GLOBAL NAVIGATION SATELLITE SYSTEM SOFTWARE DEFINED RADIO

THESIS

Jason McGinthy, Captain, USAF

AFIT/GE/ENG/10-17

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/ENG/10-17

GLOBAL NAVIGATION SATELLITE SYSTEM SOFTWARE DEFINED RADIO

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Jason McGinthy, BS

Captain, USAF

March 2010

AFIT/GE/ENG/10-17

GLOBAL NAVIGATION SATELLITE SYSTEM SOFTWARE DEFINED RADIO

Jason McGinthy, BS
Captain, USAF

Approved:

//signed//	March 2010
John F. Raquet (Chairman)	Date
//signed//	March 2010
Michael J. Veth (Member)	Date
//signed//	March 2010
Michael A. Temple (Member)	Date

AFIT/GE/ENG/10-17

Abstract

The GNSS world is quickly growing. The United States' GPS, the European Union's Galileo, China's Compass, and Russia's GLONASS systems are all developing or modernizing their signals, and there will soon be more navigation satellites in space than ever before. The goal of this research was to develop an initial capability for an AFIT GNSS software receiver. This software receiver is intended to be used for research purposes at the Advanced Navigation Technology (ANT) Center. First a GPS-only software receiver was built. It successfully acquired, tracked, and provided reasonable position estimates. Next, the receiver was successfully modified to acquire and track a Compass satellite. This only required relatively small changes in the receiver software. During the tracking process, an interesting finding was discovered concerning the secondary code structure. There are in fact two secondary codes that the transmitter alternates. After AFIT's software receiver was configured properly, the signal was successfully tracked. Finally, the receiver was modified to track one of Galileo's satellites, GIOVE-A. After correct parametric changes were made, successful acquisition and tracking of the GIOVE-A signal was accomplished. AFIT's GNSS software receiver was shown to provide a high degree of flexibility and accuracy in acquiring and tracking GNSS signals.

Table of Contents

List of Figures

Figure Page

Figure Page

List of Tables

List of Abbreviations

Abbreviation		Page
GPS	Global Positioning System	1
GNSS	Global Navigation Satellite System	1
AFIT	Air Force Institute of Technology	1
SDR	Software Defined Radio	1
RF	Radio Frequency	5
ASIC	Application Specific Integrated Circuit	6
DSP	Digital Signal Processor	6
FPGA	Field Programmable Gate Array	6
DSSS	Direct-Sequence Spread Spectrum	8
CDMA	Code Division Multiple Access	8
FDMA	Frequency Division Multiple Access	8
PRN	Psuedorandom Noise	8
LSFR	Linear Shift Feedback Register	9
IF	Intermediate Frequency	12
FFT	Fast Fourier Transform	17
DFT	Discrete Fourier Transform	17
PLL	Phase Lock Loop	18
DLL	Delay Lock Loop	18
FLL	Frequency Lock Loop	18
NCO	Numerically Controlled Oscillator	19
PIT	Pre-Detection Integration Time	23
TRIGR	Transform-Domain Instrumentation GNSS Receiver	24

GLOBAL NAVIGATION SATELLITE SYSTEM SOFTWARE DEFINED RADIO

I. Introduction

1.1 Overview

The NAVSTAR Global Positioning System (GPS) has been the dominant Global Navigation Satellite System (GNSS) for the last few decades. GPS is not only used for guiding vehicles around unfamiliar locations, it also helps put bombs on target while allowing minimal collateral damage. However, newer GNSS systems are beginning to be tested from all over the world. GPS is also modernizing its satellites and adding new signals, taking advantage of newer signal schemes that have been developed since the launch of the first block of GPS satellites. With the addition of these new signals and other GNSS developments, the skies above us are becoming ever more saturated with signals. From a research point of view, it is important to understand these signals, and to have a flexible receiver design that can be easily modified to receive multiple GNSS signals.

1.2 Problem Statement

The goal of this research was to bring the advanced capability for acquiring, tracking and decoding navigation data to the Air Force Institute of Technology (AFIT). This was accomplished by creating a Software Defined Radio (SDR), or more commonly referred to as a software receiver in the GNSS community, to allow post-processing of collected data segments in order to ensure proper acquisition, tracking, and possible decoding of signals. This receiver was built in-house from scratch except for code provided from Ohio University to import the data collection files. This receiver is intended for research purposes only, and we are not proposing that it be developed into an operational military capability.

1.3 Scope

The software receiver was initially only configured for one frequency range: 1557.5 - 1581.5 MHz. This was due to the fact that the receiver front-end does not have a large enough bandwidth to encompass all of the signals. While only one frequency range is being collected, GPS, Compass and Galileo signals all fall within the collection range. This software receiver was not configured to study the GLONASS signals, not only due to the frequency band being studied, but also due to the fact that GLONASS uses a different signal modulation scheme. This will be discussed further in Section 2.4.2. Finally, two data collections were studied with both sets collected by the same front-end (described in the following section). Data Set 1 was collected at approximately 1233 UTC on January 22, 2010, and Data Set 2 was collected at approximately 1305 UTC that same day.

The receiver is currently designed to only post-process the collected data. This post-processing allows easier initial implementation and easier modification. Currently there is no requirement for real-time processing, but this will be studied in future implementations.

1.4 Materials and Equipment

The GNSS software receiver is composed of two parts: a physical hardware front-end and the software coded receiver processer. The front-end is designed and built by the Avionics Engineering Center at Ohio University. Figure 1.1 is a picture of a 4-channel front-end. When delivered to AFIT, the hardware will include an 8-channel receiver that enables multiple frequency bands to be studied. At the time of this research, the front-end has not been delivered to AFIT, so all of the data sets were collected at Ohio University and sent to AFIT via DVDs.

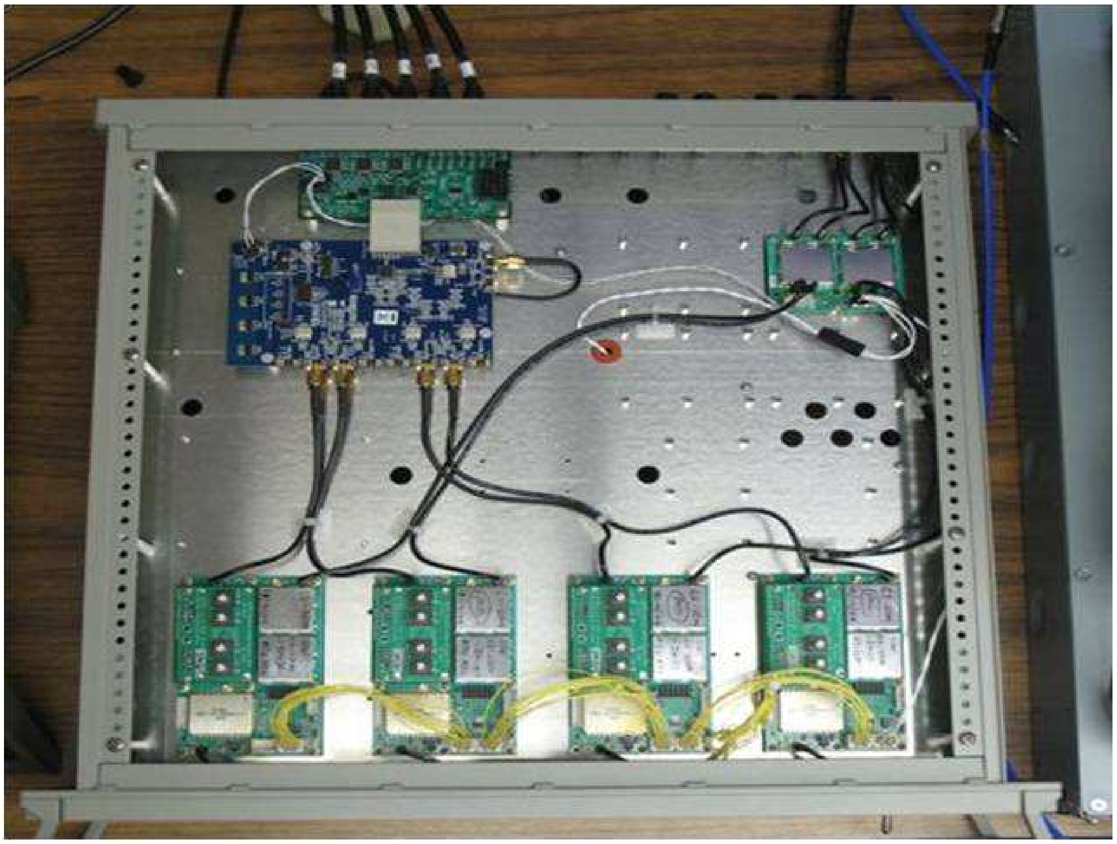


Figure 1.1. Ohio University Receiver Front-End

1.5 Thesis Organization

Chapter 2 details the background of GNSS receivers including a basic representation of how a software receiver works. It also explains the front-end that was used for data collection supplied by Ohio University. Chapter 3 details how the GPS portion of the software receiver was developed. It also provides results and analysis of collected signals. Chapter 4 details the modifications made under this research to allow the GNSS software receiver the capability to track Compass and Galileo signals.

The results and analysis of these signals are also presented in this chapter. Finally, Chapter 5 offers conclusions and recommendations of furthering this research.

II. GNSS Software Defined Radio Theory

2.1 Overview

This chapter will begin by providing background information on GNSS receivers. Then the current GNSS systems will be compared to show the similarities and differences. Next, a general description of how a software receiver works will be outlined. This will entail a detailed description of both the acquisition and tracking procedures. Finally, a section about the Ohio University front-end will be provided.

2.2 Software Defined Radio

The concept of a SDR has been around for quite some time [22]. In a SDR, Hardwired analog components of traditional receivers are replaced with digital components in order to allow programmable components that allow for greater flexibility. The SDR allows a user to change the software to quickly modify how it operates. For example, by changing a line of code, the frequency spectrum can be shifted in order to look for new signals. This basic idea allows for a flexible Radio Frequency (RF) device design. In the case of GNSS, this will allow the ability to adapt to the growing number of signals and systems in our skies. The SDR or software receiver is a highly flexible system that allows for tracking of not only GPS, but also the European Union system, Galileo, the Russian system, GLONASS and the new Chinese system, Compass.

2.3 GNSS Receiver Background

2.3.1 GNSS Hardware Receivers.

GNSS hardware receivers have been the preferred receiver implementation due to their lower cost and production in mass quantities. However, these receivers are designed with the intention of tracking a particular set of signals. These specifications do not allow for future flexibility. For instance, if a receiver is built only to use GPS, a whole new receiver would need to be built in order to accommodate a GPS and Galileo receiver. This is due to the fact that the receivers perform their acquisition and tracking loops using specific hardware, called Application Specific Integrated Circuit (ASIC)s. Since each GNSS uses different frequencies and codes, new hardware would be needed for each new capability that was needed by the user. Although a hardware receiver can have a firmware update, the whole device may need to be updated compared to the ability of being changed on-the-fly like an SDR. The inflexibility of the hardware receivers is one of the main motivation for the GNSS software receivers [15]. From strictly a militarily viewpoint, the speed in which a SDR can be modified to adapt to new threats can be invaluable.

2.3.2 GNSS Software Receivers.

GNSS software receivers are gaining more popularity due to their flexibility. Software receivers are ideal for and mainly used in research and development due to this adaptability. Many software receivers have utilized Digital Signal Processor (DSP)s, Field Programmable Gate Array (FPGA)s, the increasing computational power of computers, or any combination of the three [24, 7]. The advantages and disadvantages of these methods are described in [17]. Unlike hardware receivers, only a software update is needed to add a new tracking capability. Many software receiver applications are written in high-level computer programming languages such as C++ and

MATLAB [6]. These programming languages allow easier and quicker changes to the receiver in order to add a new GNSS signal for acquisition and tracking, in comparison to hardware-based approaches. More information on these signals will follow in Section 2.4. The actual description of how software receivers acquire and track the different GNSS satellites will be explained in Section 2.5.

2.3.3 Example of Current Software Receivers.

Software receivers are generally used in the academic world due to the research value they can add. The greater flexibility and computing powers of university programs can allow for extensive GNSS studying. Stanford University has published numerous articles on the GNSS research they have done, such as acquiring and tracking Compass and Galileo satellites [9, 8, 10]. Ohio University is also another large program dealing with GNSS research. They have developed their own GNSS receiver and published articles dealing with receiver techniques, processing schemes and other GPS related items [15, 16]. Many other academic research groups have also been working on GNSS software receivers such as the University of Calgary and Cornell University [7, 4].

2.4 Current GNSS Systems & Signals

The four current GNSS systems–GPS, Galileo, GLONASS, and Compass–generally provide positioning information in the same manner. Currently, only GPS and GLONASS are operational. However, Galileo and Compass have test satellites in orbit that broadcast signals [6, 9]. There are fundamental differences for each system, such as frequency bands, multiple access techniques, and spreading code structures. The following sections will highlight the similarities and differences of the systems.

2.4.1 Frequency Bands.

Figure 2.1 summarizes the frequency allocations of the current GNSS systems, and more detail is given in Table 2.1 through Table 2.4. This frequency range is part of the Ultra High Frequency (UHF) portion of the RF spectrum. As shown, GPS has three frequency bands: L1, L2, and L5. Galileo uses four bands: E2-L1-E1, E5A, E5B, and E6. GLONASS uses two bands: G1 and G2. Compass transmits its signals in four bands: E1, E2, E5B, and E6. GPS, Galileo, and Compass transmit signals at a single frequency in each specific band to implement Direct-Sequence Spread Spectrum (DSSS) Code Division Multiple Access (CDMA). However, GLONASS currently transmits at multiple frequencies in its bands because it was originally designed using a Frequency Division Multiple Access (FDMA) technique.

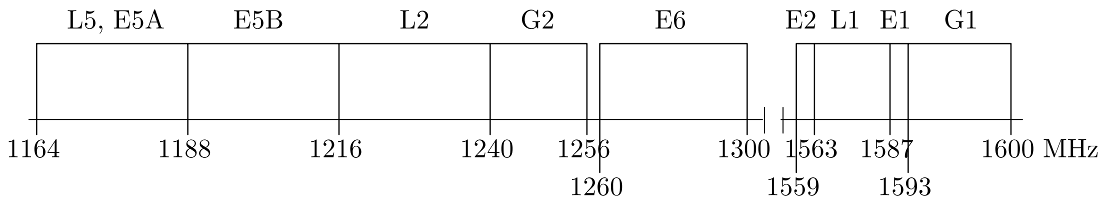


Figure 2.1. RF Frequency Allocation Chart

2.4.2 Signal Distinction Techniques.

Since each GNSS systems has multiple satellites, signal discrimination techniques are used in order to distinguish satellites from one another. As mentioned previously in Section 2.4.1, two types of modulation are used in GNSS systems. First, GPS, Galileo, and Compass use DSSS CDMA. CDMA allows all satellites in a particular GNSS system to transmit at the same frequency. In order to distinguish the satellites, Psuedorandom Noise (PRN) spreading codes are used. Each satellite has a unique PRN. The design of these codes will be further explained in Section 2.4.3. These codes are known by the receiver, which allows it to acquire and track the specific satellites.

Second, GLONASS currently uses the FDMA technique. This type of signaling scheme distinguishes each satellite by carrier frequency. For example, for signals being transmitted in the GLONASS G1 band, the center frequency is 1602 MHz and the other channels are separated at 0.5625 MHz spacings. GLONASS also uses a PRN code in order to enable spread spectrum at each frequency. In GLONASS, all satellites transmit the same PRN code. However due to FDMA, the PRN is transmitted in the different frequency channels. Two satellites share one channel, but the satellites are located on opposite sides of the earth in order for a receiver to only have one satellite in view [21]. In the near future, GLONASS will begin to use CDMA technology in order to be more easily integrated into multi GNSS system receivers [11].

2.4.3 Spreading Code Structure.

The current GNSS systems have similarities and differences in the structure of the spreading codes used to distinguish the satellites. These spreading codes also reduce the threat of jamming due to the signal being spread over a specified bandwidth. The codes are used in the acquisition and tracking processes of the receiver which are described in Sections 2.5.1 and 2.5.2. Currently, GPS uses Gold codes of length 1023 chips for its C/A signals. Gold codes are PRN spreading codes formed from 2 Linear Shift Feedback Register (LSFR)s. All the C/A codes for GPS are generated by changing the phase of the second LSFR [6]. GPS uses a 10-stage Gold code for each C/A code. This code is repeated every 1 ms, which enables quick satellite acquisition. GPS is adding new signals with longer codes of 10,230 and 767,250 chips on the new L2 band and 10,230 chips on the new L5 band. However, these new codes are not Gold codes but are formed from LSFRs. These longer codes decrease the sidelobes of the autocorrelation and crosscorrelation outputs so there will less of a chance for interference from other satellites [21]. The spreading code is mixed with the 50 bps GPS data message and transmitted on a carrier frequency corresponding to the frequency band. It is to be noted that some of the new signals do not have the same 50 bps data message mixed with them [21].

Galileo, although not yet fully operational, uses similar code structures to that of GPS. For example, in the L1 band, Galileo uses a spreading code that is 4092 chips and repeats every 4 ms. This code is generated by a truncated Gold code sequence. Once 4092 chips are reached, the LSFR is reset to its initial state. There is also a secondary code of 25 chips that modulates the 25 specific repetitions of the primary 4092 code to form a tiered code of length $4092 \times 25 = 102,300$ chips [6]. The data message on the L1-B data channel is transmitted at 250 bps [3].

Recently, the structure of Compass codes have been studied by [9] and [8]. It appears that the signals transmitted on the E2 band are created from an 11-stage Gold code. It consists of 2046 bits with a 1 ms repeat interval. Compared to GPS, the Compass E2 PRN code has twice as many chips in the same interval, which equates to a chipping rate of 2.046 Mchips/sec. Also it appears that this same signal is being transmitted on E5B. Also on the E2 band, there is a secondary 20 bit Neuman Hoffman code repeated at a 1 KHz rate. Also the code at E6 appears to be generated from two 13-stage Gold codes [9].

Although GLONASS employs a FDMA scheme to distinguish satellites, a 511 chip PRN code is transmitted on the G1 band. Only one code is needed since the satellites are distinguished by the channel frequency. For more precision a 511,000 chip code is broadcast on both G1 and G2 bands [21]. Tables 2.1, 2.2, 2.3 and 2.4, show a quick summary of some of the characteristics of the current GNSS systems.

Table 2.1. Current GPS Signal Characteristics (Civil Signals) [21]

Signal Name	*Center Frequency*	*Code Length & Chipping Rate*
L1 C/A	1575.42 MHz	1023 chips 1.023 Mcps
L2	1227.40 MHz	10,230 chips (CM) 767,250 chips (CL) 1.023 Mcps
L5	1176.45 MHz	10,230 chips 10.023 Mcps

Table 2.2. Current Galileo Signal Characteristics (Open Service Signals)[6, 21]

Signal Name	*Center Frequency*	*Code Length & Chipping Rate*
E2-L1-E1 OS	1575.42 MHz	4092 chips (primary) 25 chips (secondary) 1.023 Mcps
E5A	1176.45 MHz	10230 chips 10.23 Mcps
E5B	1207.14 MHz	10230 chips 10.23 Mcps

Table 2.3. Current Compass Signal Characteristics [9, 8]

Signal Name	*Center Frequency*	*Code Length & Chipping Rate*
E1	1589.74 MHz	N/A
E2	1561.098 MHz	2046 chips (CL) 2.046 Mcps
E6	1268.52 MHz	10,230 chips (I-chan) 10.023 Mcps
E5B	1207.14 MHz	2046 chips 2.046 Mcps

Table 2.4. Current GLONASS Signal Characteristics[21, 2]

Signal Name	*Frequency Band*	*Code Length & Chipping Rate*
G1	1602 MHz	511 chips (SPS) 0.511 Mcps 511K chips (PPS) 5.11 Mcps
G2	1246 MHz	511K chips (PPS) 5.11 Mcps

2.5 Receiver Signal Processing

Since GPS is the only operational system using CDMA technology, it will be the main example for the following sections. Figure 2.2 shows a basic diagram of a GNSS receiver. The signal arrives at the antenna and through a series of mixers and filters is downconverted to a manageable Intermediate Frequency (IF). Then it is converted into a digital signal containing two components, the in-phase portion, I, and the quadrature portion, Q, through the RF front end. The sampled in-phase signal is represented as

$$I_k = \frac{A}{\sqrt{2}} C_k D_k \cos(\Delta\omega_B t_k + \phi_0), \tag{1}$$

where C_k is the signal's associated Gold code, D_k is the GPS data message, and $\Delta\omega_B$ is the baseband frequency plus the Doppler frequency in radians/sec [27]. The Doppler frequency is due to the combined effects of relative motion between the satellite and the receiver and clock drifts in the satellite and receiver. Throughout this thesis, the Doppler Shift shall be negative as the satellite is moving towards the receiver and positive as it begins to move away from the receiver. The sampled phase of the signal is represented by the following equation:

$$\phi_k = \Delta\omega_B t_k + \phi_0. \tag{2}$$

Therefore, substituting Equation 2 into Equation 1,

$$I_k = \frac{A}{\sqrt{2}} C_k D_k \cos(\phi_k). \tag{3}$$

The sampled quadrature signal is similarly expressed as

$$Q_k = \frac{A}{\sqrt{2}} C_k D_k \sin(\phi_k). \tag{4}$$

Finally, the signal components go into the acquisition and tracking algorithms of the receiver. The antenna and RF front end will not be discussed any further since the acquisition and tracking loops are the main focus of a software GNSS receiver. Once the satellite signal is acquired, the navigation data can be decoded from the signal. Sections 2.5.1 and 2.5.2 will explain in detail the acquisition and tracking processes of a GNSS software receiver.

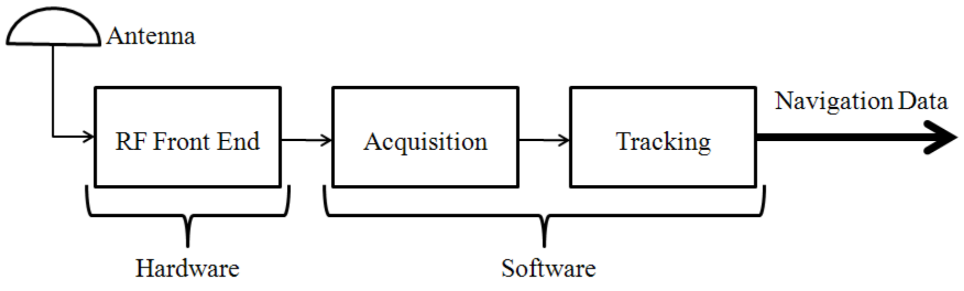


Figure 2.2. Basic GNSS Software Receiver Overview Diagram

2.5.1 Satellite Acquisition.

In order for a receiver to use information gathered from any of the GNSS satellites, the satellite signal must be acquired. The signal must be acquired in order to pass the appropriate parameters to the tracking portion of the receiver. More information about satellite tracking will be discussed in Section 2.5.2. Figure 2.3 shows the acquisition process performed by a receiver.

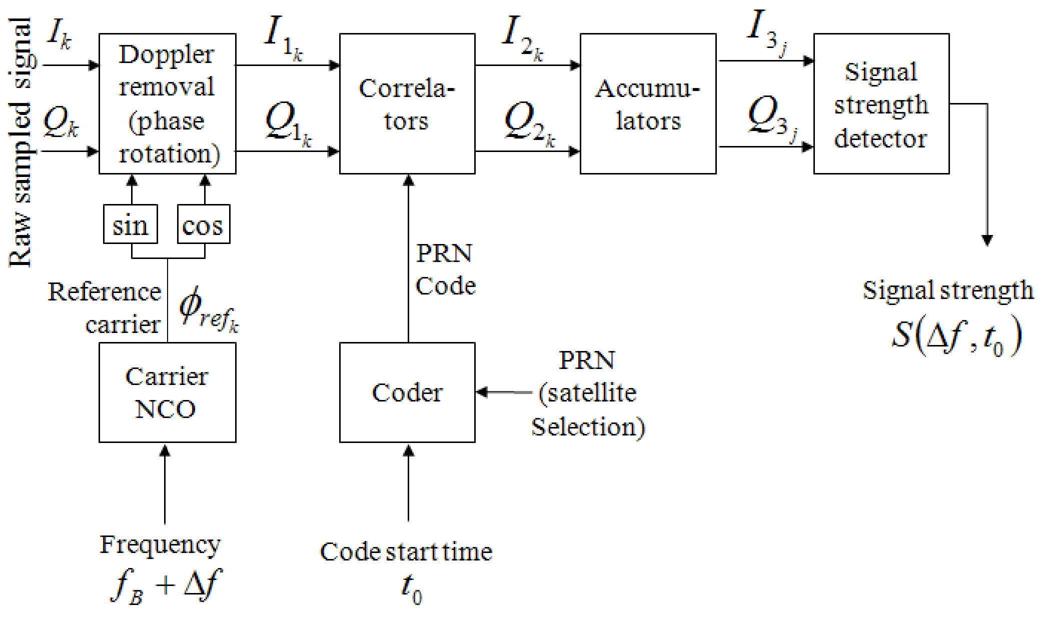


Figure 2.3. Satellite Acquisition Data Flow[25]

The receiver may have no *a priori* information about which satellites are visible at the moment it starts. This is called a "cold start," since the receiver must start randomly trying to acquire satellite signals. The receiver has no knowledge of its position or time since the receiver has no information from the GNSS system's almanac. A "warm start" occurs when the receiver has a reasonable estimate of its position or time and has a reasonably current almanac [21]. This is possible if the receiver was turned off and turned on only a short amount of time later. In this case, the receiver has a recent, accurate almanac and ephemeris data. It is able to search for and acquire satellites more quickly since it knows which satellites should be located in the sky and what the appropriate Doppler shifts should be.

In order for a receiver to acquire a satellite, it must have the proper PRN codes stored in its memory so they can be generated in order to match a satellite's PRN code. This process utilizes the quasi-orthogonality benefits of Gold codes in order to keep cross-correlation values much lower than the auto-correlation value of a specific PRN code of a specific satellite. This allows for a very low probability of locking onto a satellite with the wrong PRN code.

In order for a receiver to acquire a specific satellite, the ranges for the timing of the PRN code and the carrier frequency plus Doppler shift must be searched. For GPS, the PRN code consists of 1023 chips, and depending on the sampling rate of the receiver, there are an infinite amount of code phases that can be searched. Generally, independent of the sampling rate, the code phase can be searched in 1/2 chip intervals allowing 2046 possible code phases. Typically, for a fast moving receiver, the Doppler Shift can range from ± 10 KHz [6]. However, for a stationary receiver, the Doppler Shift range is approximately ± 4 KHz. Nonetheless, in order to properly acquire an incoming satellite signal, a very large number of samples need to be evaluated depending on the resolution required. The receiver will only lock onto a satellite if the correct frequency and code phase are both found. The techniques for the acquisition portion of the receiver will be described in Sections 2.5.1.1 and 2.5.1.2. The method in which a satellite is determined to be acquired or not is based on the search output signal strength and will be discussed in Section 2.5.1.3.

2.5.1.1 Serial Search Method.

The serial search method searches each combination of frequency and code timing in a sequential manner. Figure 2.4 is a diagram demonstrating the two dimensional search that much be accomplished. Equation 5 gives a "rule of thumb" relationship between the integration time and the size of each Doppler bin [29],

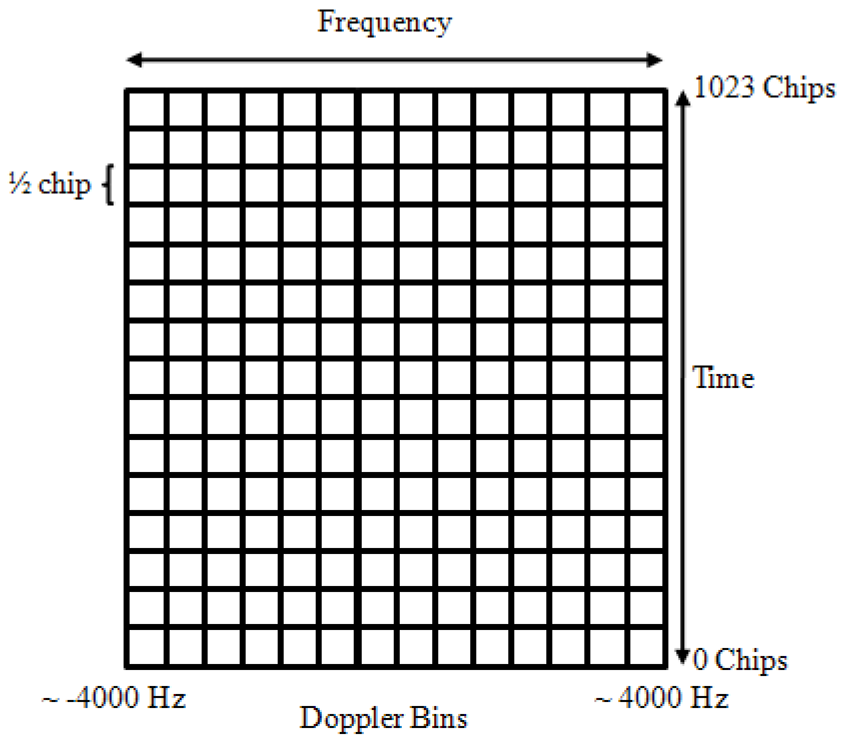


Figure 2.4. 2-Dimensional Acquisition Search Diagram

$$\text{Doppler Bin Size} = \frac{2}{3T} \tag{5}$$

where T is the integration time. If the 8 KHz frequency range is divided into Doppler bins with sizes of 667 Hz (typical for 1 ms integration time), 12 frequency bins are created, totaling $12 \cdot 2046 = 24,552$ different combinations that must be searched for each satellite PRN code. The incoming I_k and Q_k signals are multiplied by each combination. The procedure starts at the lowest frequency bin (or, if there is a guess at the Doppler, a bin which is near that guess) and zero time offset and continues through each 1/2 chip interval and then proceeds to the next bin. It is important for each Doppler bin to begin at the 0 chip location in order to reduce multipath interference since the first signal to arrive is typically the original. This method is very simple to implement since it just cycles through each combination sequentially. This is typically how a hardware receiver performs the satellite acquisition process. However, this method can be quite slow and software receivers are capable of running faster algorithms which will be discussed in the next section.

2.5.1.2 Fast Fourier Transform Method.

The Fast Fourier Transform (FFT) method has been used since the early 1990s [28], and is a common method in many software GNSS receivers [5, 12, 15, 18, 19, 20, 23, 24, 26]. This method is similar to the serial method mentioned previously, but instead of searching each individual bin one at a time, it is able to search all chip offsets in one Doppler bin simultaneously, greatly reducing the search time. The FFT method can be used to search all code phase bins in one operation but still must increment through the frequency bins. However, for this method to be most efficient, the sampled signal and codes should be of length 2^n, where n is a positive integer number. A similar method to the FFT is the Discrete Fourier Transform (DFT) which does not require lengths of 2^n. For the case of GPS, the code length is 1023 chips, so an extra zero is added to the end of the code to perform the radix-2 FFT. Therefore, the search time is drastically reduced, since the FFT can be performed very quickly by today's processors.

2.5.1.3 Satellite Detection.

From the outputs of the search algorithm, the signal correlation power is determined by the expression $\sqrt{I^2 + Q^2}$, or a similar variation. Independent of which method is used, the satellite is considered acquired if the power level exceeds a set threshold. This threshold will only be exceeded when the correct frequency and code timing are found. Figure 2.5 is an example of the output of a code phase and Doppler search space. The largest peak indicates the correct Doppler shift and code phase offset. For a given threshold value of 6×10^4, the satellite is detected and the signal is acquired.

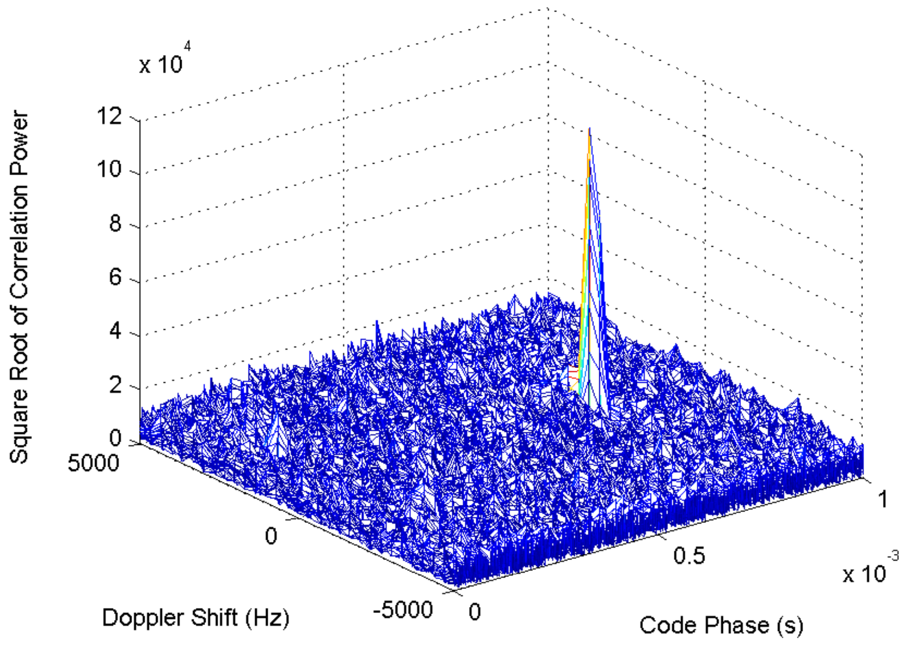


Figure 2.5. Example of Acquisition Search Output

2.5.2 Satellite Tracking.

Once a satellite is detected and signal acquired, the tracking process begins in order to keep a lock on the satellite since it is continuously moving. There are two main tracking loops that are used for this process: Phase Lock Loop (PLL) for carrier-phase tracking and Delay Lock Loop (DLL) for code tracking. A third loop called a Frequency Lock Loop (FLL) is normally used in the acquisition portion, but can also be used as a tracking loop and is very similar to a PLL. These loops must work together for the receiver to track properly. Figure 2.6 illustrates the tracking portion of the receiver. The next sections will describe each loop, but it will be assumed that the other loops other than the one being described are working perfectly in order to simplify the explanation.

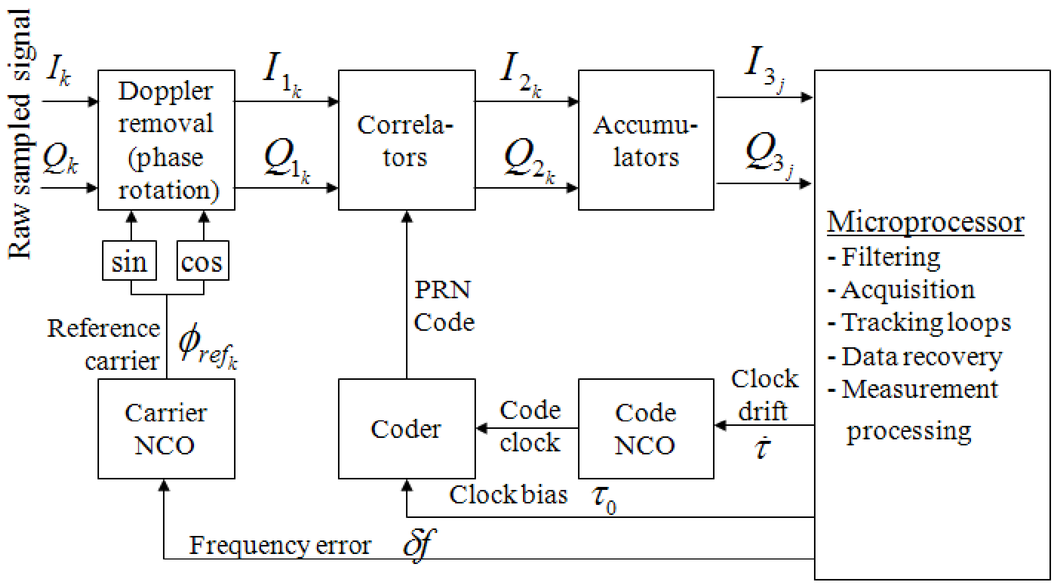


Figure 2.6. Satellite Tracking Loop[25]

2.5.2.1 Phase Lock Loop.

The PLL tracks the phase of the incoming signals with the locally generated code. This procedure is also referred to as "Doppler Removal" because the Doppler frequency component of the signal is ideally removed so that only the codes remain at the desired baseband frequency. This process occurs when a carrier Numerically Controlled Oscillator (NCO) mixes a reference frequency with the incoming signal. If the frequencies are identical, then the carrier frequency goes to baseband. This could also be called "Carrier Removal" since the actual carrier frequency is removed. Once the frequencies are identical all that is left to match are the phases of the signals. To better illustrate this, the I_k and Q_k signals defined by Equations (3) and (4) need to be examined. Figure 2.7 shows a phasor diagram of the total signal that is composed of these two composite signals.

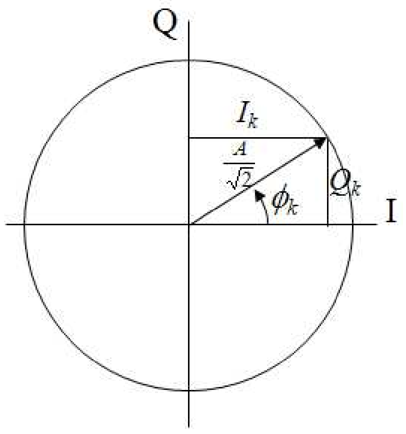


Figure 2.7. Phasor representation of incoming GPS signal[25]

The phase of the signal as defined in Equation (2) is the main focus in this diagram. If the incoming signal and receiver generated signal have the same frequency and phase, then the mixed signal will have all the power in the in-phase portion of the signal. This is illustrated in Figure 2.8.

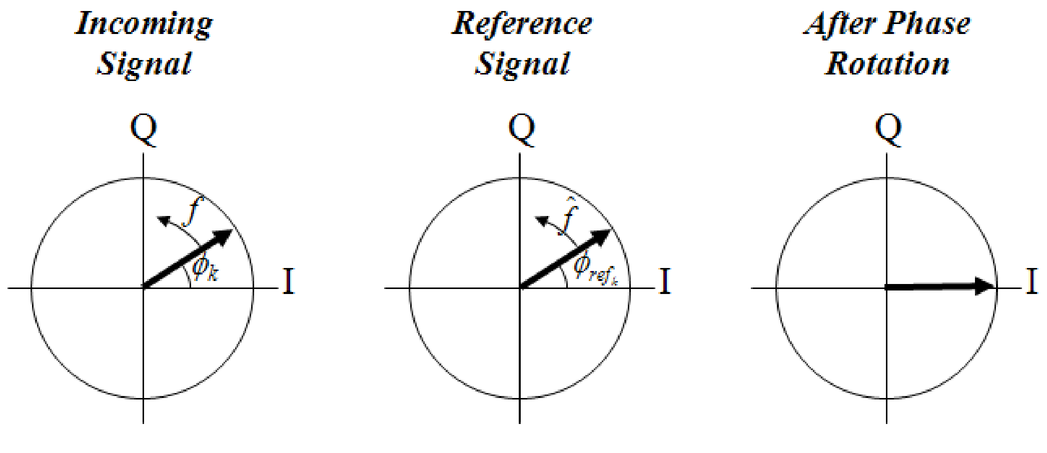


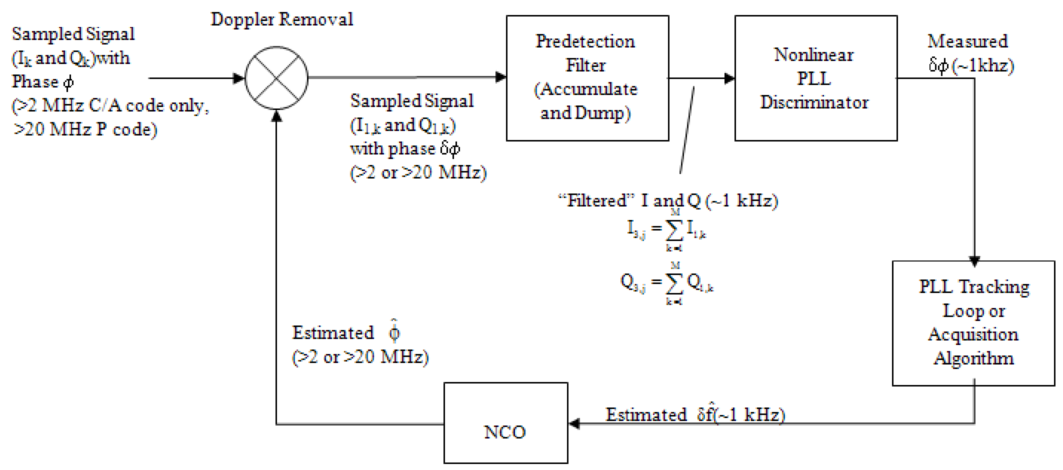
Figure 2.8. PLL Phasor example [25]

The output I and Q samples are then accumulated for a period of time typically ranging from 1 ms to 20 ms and then used in a PLL discriminator algorithm. Table 2.5

Table 2.5. Phase Lock Loop Discriminators [29]

Algorithm	*Costas Loop*	*Output Phase Error*
$\text{sign}(I) \cdot Q$	Yes	$\sin \phi$
$I \cdot Q$	Yes	$\sin 2\phi$
Q/I	Yes	$\tan \phi$
$\text{atan}(Q/I)$	Yes	ϕ
$\text{atan2}(Q, I)$	No	ϕ

shows a list of PLL discriminators. When dealing with phase, the possibility of a data bit transition may occur and the PLL discriminator should be insensitive to this event. A tracking loop which uses a discriminator that is insensitive to the 180° phase flip is a Costas loop [29]. The PLL discriminator evaluates the PLL outputs and determines the phase offset that needs to be sent to the carrier NCO. This total process is illustrated in Figure 2.9.



Sampled Signal (I_k and Q_k)with Phase ϕ (>2 MHz C/A code only, >20 MHz P code)
Doppler Removal
Sampled Signal ($I_{1,k}$ and $Q_{1,k}$) with phase $\delta\phi$ (>2 or >20 MHz)
Predetection Filter (Accumulate and Dump)
Nonlinear PLL Discriminator
Measured $\delta\phi$ (~1khz)
"Filtered" I and Q (~1 kHz)
$I_{3,j} = \sum_{k=1}^{M} I_{1,k}$
$Q_{3,j} = \sum_{k=1}^{M} Q_{1,k}$
PLL Tracking Loop or Acquisition Algorithm
Estimated $\hat{\phi}$ (>2 or >20 MHz)
NCO
Estimated $\delta\hat{f}$(~1 kHz)

Figure 2.9. Detailed view of a Phase Lock Loop [25]

2.5.2.2 Frequency Lock Loop.

The FLL is very similar to the PLL previously described. It is a simpler process since it is only tracking the frequency and not the phases of the signals. When operating properly (i.e., in frequency lock), the frequency difference between incoming and reference signals will go to zero, but the phase difference will be a constant value. Figure 2.10 shows a phasor representation of this process. The FLL is typically used in the acquisition portion of the signal processing since it isn't needing the precise phase that tracking requires. Then once the frequency is determined the PLL begins tracking the signals. The FLL is also advantageous when the system is in a high dynamic situation since it is more robust to dynamics [29].

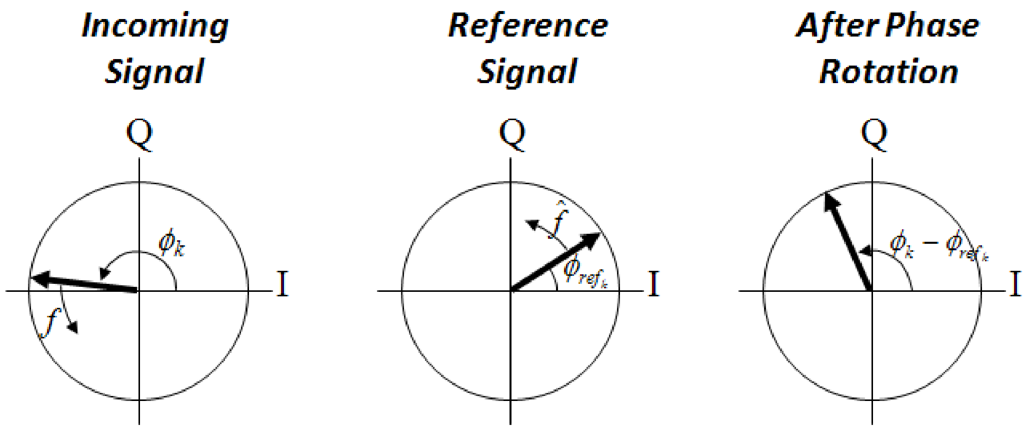


Figure 2.10. FLL Phasor example [25]

2.5.2.3 Delay Lock Loop.

The DLL tracks the timing of the received code with a locally generated code in order to keep lock on the satellite. Since the sampled I_k and Q_k signals are perfectly in phase with the receiver's frequency generator, only the PRN code and data code portions of the sampled signals remain. The DLL relies on the autocorrelation of the incoming PRN code and the NCO generated PRN code. The definition of autocorrelation for a function, $f(t)$, is

$$R(\tau) = \int_{-\infty}^{\infty} f(t)f(t+\tau)dt. \tag{6}$$

The coder produces three versions of the local PRN code which are early, prompt, and late copies. For example if the chip spacing of the correlators is 1, the early copy is advanced 1/2 chip of the expected code, the prompt is an exact replica of the expected code, and the late copy is delayed by 1/2 chip. Then incoming sampled I_k and Q_k signals are multiplied by the three variants of the local PRN code and produce six correlator outputs, I_E, I_P, I_L, Q_E, Q_P and Q_L. When perfectly tracking (with zero timing error), the magnitude of the early and late correlator outputs will be equal to each other and the prompt correlator normalized output will be 1. These outputs are then sent through accumulators in order to strengthen the carrier-to-noise ratio.

Then the accumulator outputs are then sent to a DLL discriminator. There are two forms of DLL discriminators, coherent and non-coherent. A coherent discriminator requires phase lock so all the power is in the I portion of the signal. Non-coherent discriminators do not require phase lock, but the performance decreases if the frequency estimate gets worse. Table 2.6 lists some coherent and non-coherent DLL discriminators. The discriminator output determines if a new timing offset is to be sent to the NCO to either speed up or slow down the code. Figure 2.11 illustrates this DLL process.

2.5.3 Pre-Detection Integration Time.

The Pre-Detection Integration Time (PIT) is the length of integration time for the correlators in the acquisition and tracking loops. Typically for GPS, the PIT starts at 1 ms since that is the length of one C/A code epoch. However, once the tracking loop is properly locked onto the signal, the PIT can be increased in order to increase signal to noise ratio. Unless special coding has been implemented, the longest PIT is normally 20 ms due to the fact that each data message bit is 20 ms long. If the PIT were to be longer than 20 ms, there is the possibility of a bit transition occurring

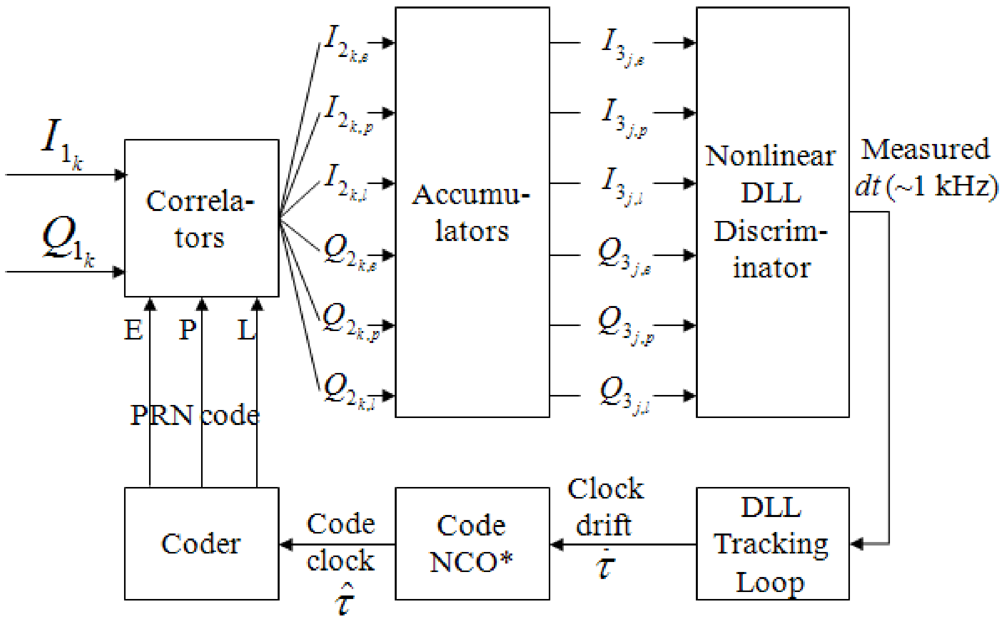


Figure 2.11. Detailed view of a Delay Lock Loop[25]

during the integration which could significantly decrease the signal to noise ratio.

2.6 Ohio University Receiver Front-End

The Ohio University Avionics Engineering Center has developed their own GNSS receiver named Transform-Domain Instrumentation GNSS Receiver (TRIGR) [16]. AFIT has purchased an 8-channel model for researching GNSS signals. Figure 1.1 shows an image of the physical hardware that receives the signal. Using the GPS L1 band and Figure 2.12 as an example, the received RF signal from the antenna is first downconverted to a 70 MHz center frequency, passed through a bandpass filter, downconverted a second time, passed through a second bandpass filter, and finally sampled at a 56.320 MHz sampling rate. (This provides a baseband sampled signal centered at 13.68 MHz.) The signal then passes through a 12-bit Analog-to-Digital converter. This digital signal is then separated into 1 ms blocks of data. The software receiver then reads in these blocks of data one at a time. Figure 2.12 shows a diagram

Table 2.6. Delay Lock Loop Discriminators [29]

Coherent
$I_E - I_L$
Non-Coherent
• Dot product power $(I_E - I_L)I_P + (Q_E - Q_L)Q_P$ • Early minus late power $\frac{1}{2}[(I_E^2 + Q_E^2) - (I_L^2 + Q_L^2)]$ • Early minus late envelope $\frac{1}{2}[\sqrt{I_E^2 + Q_E^2} - \sqrt{I_L^2 + Q_L^2}]$ • Normalized early minus late envelope $\frac{\sqrt{I_E^2+Q_E^2}-\sqrt{I_L^2+Q_L^2}}{\sqrt{I_E^2+Q_E^2}+\sqrt{I_L^2+Q_L^2}}$

of how this process works inside the receiver's front-end.

2.7 Summary

This chapter described the background of SDRs and software receivers. The comparison of hardware and software receivers was also presented. Some examples of groups that are using software receivers was provided. Next, the major GNSS systems were compared to provide insight into the receiver modifications needed to encompass all of these signals. The next sections then explained how a software receiver works. Finally, the Ohio University provided front-end was described to explain how the signal is manipulated from the antenna to the actual digital sample outputs.

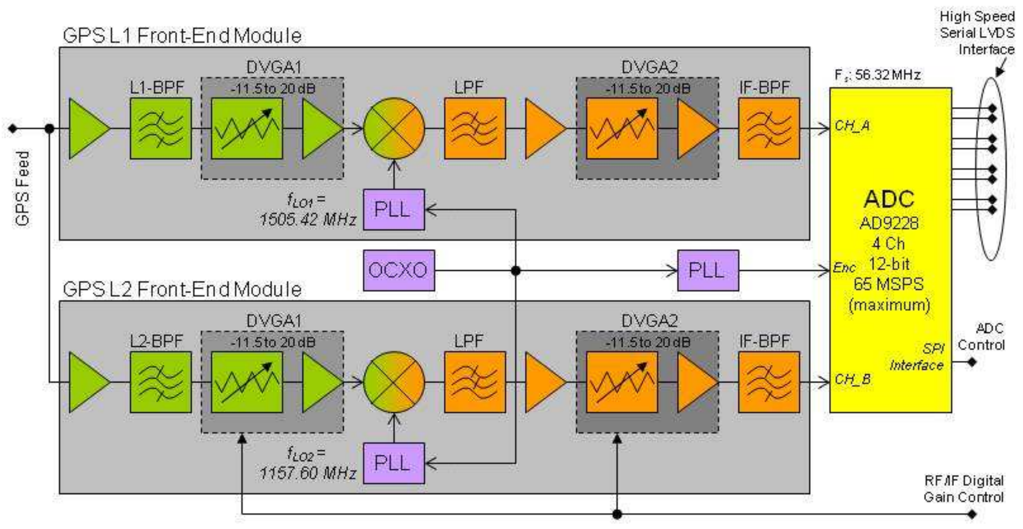


Figure 2.12. Ohio University Receiver Front-End Diagram[16]

III. GPS-Only Software Defined Radio

3.1 Overview

This chapter describes the development of the AFIT GNSS SDR. First, a GPS-only receiver was designed. This chapter will discuss this process including such topics as the overall system process flow, PRN generation, target acquisition and tracking, post-tracking calculations, data message decoding, and position estimation. Chapter IV will describe the modifications made to the GPS receiver code to incorporate Compass and Galileo. The results for these systems will be are presented there as well.

3.2 Overall System Process Flow

In order to determine what satellite signals are in the data collections, the receiver was first set to an acquisition-only mode. All 32 GPS satellites were searched for, and the found satellites were then used in the rest of the receiver. This allowed quicker processing once the known satellites were only being tracked. Next, each satellite signal was individually run through the receiver to be tracked and the outputs were saved. After all the satellites were tracked, the stored outputs for pseudorange calculations were input into the pseudorange estimation procedure, which is explained in Section 3.7.3, to determine synchronized pseudoranges. The pseudoranges were then stored in a file, which subsequently was used in the position estimation procedure, which is described in Section 3.7.4. Figure 3.1 illustrates the processes flow.

3.3 GPS PRN Generation

In order to generate the PRNs for the GPS satellites, the LSFR structure must be known. There are many GPS C/A code generators available, since the information

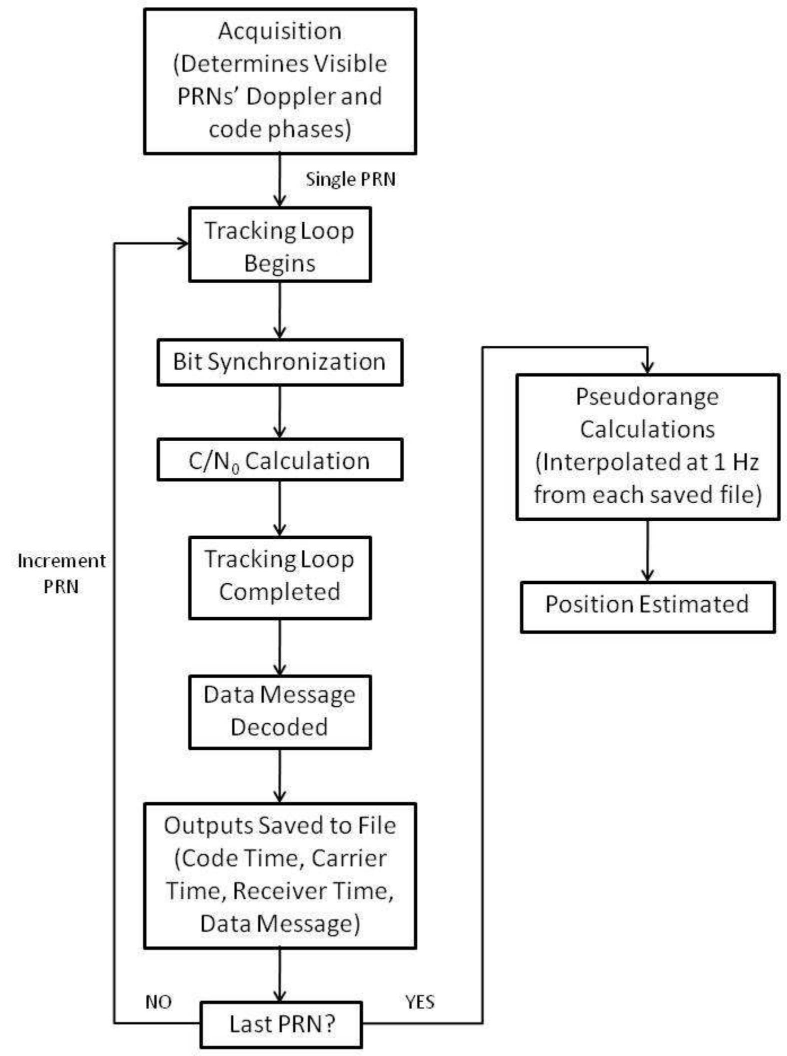

Acquisition (Determines Visible PRNs' Doppler and code phases)
Single PRN
Tracking Loop Begins
Bit Synchronization
C/N0 Calculation
Tracking Loop Completed
Data Message Decoded
Outputs Saved to File (Code Time, Carrier Time, Receiver Time, Data Message)
Last PRN?
NO
YES
Increment PRN
Pseudorange Calculations (Interpolated at 1 Hz from each saved file)
Position Estimated

Figure 3.1. Software Receiver Flow Diagram

is open to the public from many textbooks and the ICD-GPS-200c [1, 6, 21]. With the known generator, a MATLAB script can be written to generate the PRNs.

3.4 GPS Acquisition

There was no prior knowledge as to which satellites were in view at the data collection time, so all satellites had to be searched for first. As discussed in Section 2.5.1, acquiring a satellite requires searching through the appropriate Doppler Shift and code phases. A coherent integration time of 1 ms with 5 non-coherent integrations was used. Therefore, 5 ms of data was sent into the acquisition process. The incoming signal was sampled at a rate of 56.320 MHz. Therefore, each millisecond block of data contained 56320 samples.

Next, every 11 samples of the sampled signal were averaged together which reduced the number of samples to 5120 and allowed a code phase spacing of 1/5 chip. A replica PRN code was locally generated and divided into the appropriate number of bins. For the particular data sets provided, a Doppler Shift range of ± 5000 Hz was searched. The FFT search algorithm was implemented in order to reduce the search time. Since there are 5 non-coherent integrations performed, the data is integrated at 1 ms PITs and then added non-coherently five times forming a more pronounced peak. This larger peak is then easier to differentiate from the noise floor of the rest of the search space. After the search, the location of the peak determined the Doppler Shift and code phase. Figure 3.2 shows the acquisition result of PRN 27. The peak is very evident compared to the noise levels.

In order to determine whether a signal is present or not, 3 consecutive sets of 5 ms of data were processed by the acquisition algorithm. For each set, the maximum peak was determined. If at least 2 of the 3 peaks were in the exact same location, it was assumed that a signal was present [13]. Once a signal is determined present, the initial frequency and code phase parameters are then fed into the tracking loop algorithm.

3.5 Tracking

With the approximate Doppler Shift and code phase of the beginning of the C/A code epoch, the tracking loop began to track the signal. The signal data is read in from the provided files using adapted code provided by Ohio University. The provided

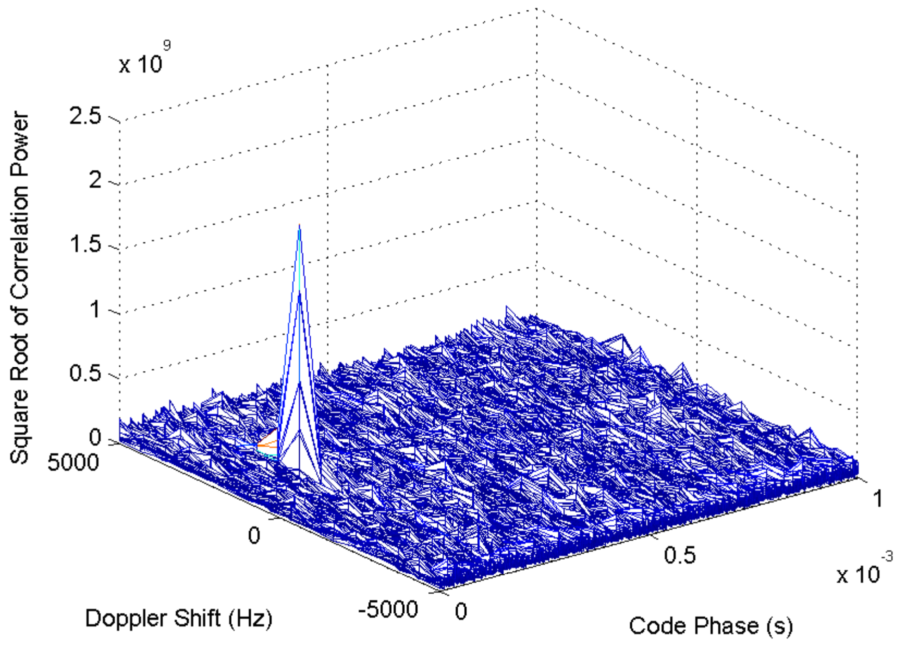


Figure 3.2. Successful Acquisition of GPS Signal for PRN 27

data is separated into 1 ms blocks of data. Since the code phase of the signal can be any of the 56320 samples, the sample delay of the signal is calculated by multiplying the code phase by the sample frequency. The data was read into a 2 ms buffer in order to allow the skipping of blocks needed for the alignment of the sample delay. The 1 ms data block correctly locating the beginning of the C/A code epoch was then output from the buffer. This process essentially initializes the C/A code delay to a value close to zero (meaning that the PRN code sequence starting at zero lines up with the incoming signal sequence).

3.5.1 Doppler Removal.

The next part of the tracking process was the removal of the carrier frequency and Doppler Shift. This was performed by mixing the incoming signal with the signal's IF of 13.68 MHz to reduce the carrier frequency to baseband. The NCO can change the actual IF based on the change in the Doppler Shift. The carrier NCO speeds up or slows down the rate at which the IF changes based on the outputs of the FLL/PLL filter. This will be discussed in further detail in Section 3.5.4. For the Ohio University data, the signal is still the real-valued components until the Doppler removal. In order to separate these components, the signal is mixed with a cosine and a sine wave at the current Doppler frequency and phase. These components are represented in Equations (7) and (8) [27].

$$I_{1_k} = \frac{A}{\sqrt{2}} C_k D_k \cos\left(\phi_k - \phi_{rk}\right) \tag{7}$$

$$Q_{1_k} = \frac{A}{\sqrt{2}} C_k D_k \sin\left(\phi_k - \phi_{rk}\right) \tag{8}$$

where ϕ_k is the current phase of the signal and ϕ_{rk} is the reference phase. After this procedure if the signal phase and reference phase are equal, the in-Phase and quadrature components, I_{1_k} and Q_{1_k}, are further reduced as depicted in Equations (9) and (10).

$$I_{1_k} = \frac{A}{\sqrt{2}} C_{1_k} D_{1_k} \tag{9}$$

$$Q_{1_k} = 0 \tag{10}$$

These new signals correspond to the same I_{1_k} and Q_{1_k} in Figure 2.6. After this carrier wipe-off, the signal continues to the correlation process.

3.5.2 PRN Code Removal.

After the Doppler removal, the PRN code is removed from the signal. As previously mentioned in Section 2.5.2.3, the incoming signal is correlated with three locally generated versions of the PRN code. These three versions are Prompt, Early, and

Late corresponding to a timing offset based on the DLL discriminator's chip spacing. The chip spacing for these correlators was set to 0.4 chips. After the correlation process was performed, the output was then summed to form one value from each correlation. MATLAB easily performs this operation by matrix multiplication as shown in Equation (11).

$$\begin{bmatrix} I_P & I_E & I_L \\ Q_P & Q_E & Q_L \end{bmatrix} = \begin{bmatrix} I_{1_k} \\ Q_{1_k} \end{bmatrix} \times \begin{bmatrix} P & E & L \end{bmatrix} \tag{11}$$

P, E, and L represent the prompt, early and late versions of the PRN code. I_{1_k}, Q_{1_k}, P, E, and L are all 1×56320 long vectors. These summed output values are labeled I_P, I_E, I_L, Q_P, Q_E and Q_L and are then sent to the discriminators as described in Section 2.5.2.3.

3.5.3 Discriminators.

The accumulated correlation outputs enter the FLL, PLL and DLL portion of the software receiver. The FLL discriminator used for this receiver is shown in Equation (12) [29].

$$FLL_{out} = \frac{ATAN2(cross, dot)}{(t_2 - t_1)2\pi} \tag{12}$$

where

$$dot = I_{P1} \cdot I_{P2} + Q_{P1} \cdot Q_{P2}$$
$$cross = I_{P1} \cdot Q_{P2} - I_{P2} \cdot Q_{P1}$$

The PLL used in this project is the $ATAN(\frac{Q_P}{I_P})$ Costas Loop discriminator from Table 2.5. This decision allowed the loop to be insensitive to the 180° phase change that occurs when a bit flips. For this receiver, the FLL is only on for the first 120 ms and then it is turned off as the PLL continues the phase tracking of the signal. The PLL does not begin tracking until after 40 ms have passed in order to allow the FLL to lock in on the right frequency. The FLL/PLL discriminator outputs are then passed to the FLL/PLL loop filter to determine the change in the Doppler Shift and allows for precise tracking of the frequency.

The DLL discriminator chosen for this receiver is the normalized early minus late envelope non-coherent discriminator. This was chosen due to the fact that the discriminator does not need to know the actual power levels from the correlator outputs ahead of time (since they are normalized). This discriminator runs the entire time that the signal is being tracked. The DLL discriminator output is then passed onto the DLL loop filter in order to determine and track the proper code phase.

3.5.4 FLL/PLL Filter.

The FLL/PLL filter design chosen for this receiver is a first-order loop for the FLL aiding the second-order loop of the PLL. Figure 3.3 displays a block diagram of the combination of these filters with the implementation of a digital bilinear transform integrator for the second-order PLL loop. The inputs to this filter are the outputs from the FLL and PLL discriminators. It must be noted that these inputs are in units of Hertz, therefore, the noise bandwidth parameters are also in units of Hertz.

The noise bandwidth (B_n) for a first-order filter is calculated by Equation (13) [29], and the B_n for a second-order filter is calculated using Equation (14) [29].

$$B_n = \frac{\omega_0}{4} \tag{13}$$

$$B_n = \frac{\omega_0(1 + a_2^2)}{4a_2} \tag{14}$$

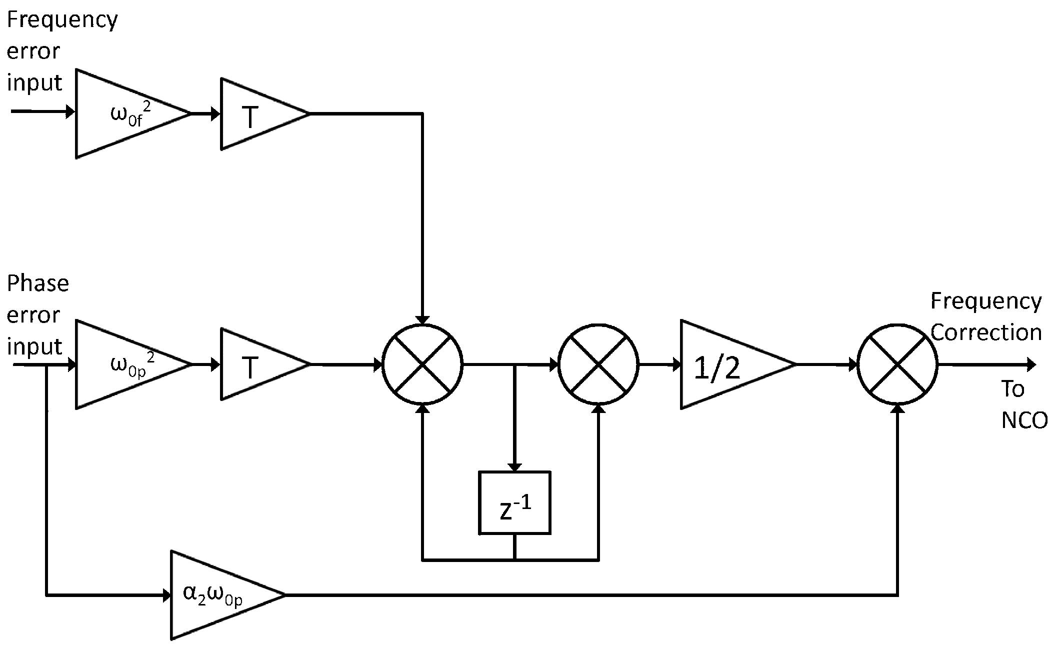


Figure 3.3. Block diagram of first order digital loop filter aiding a second order digital filter implemented with a digital bilinear transform integrator adapted from[29].

where

$$a_2 = \sqrt{2}$$

Normally, a B_n is set and then ω_0 is calculated using either Equation (15) for a first-order loop or Equation (16) for a second-order loop [29].

$$\omega_0 = \frac{B_n}{0.25} \tag{15}$$

$$\omega_0 = \frac{B_n}{0.53} \tag{16}$$

The steady state error for the first-order filter is $\frac{\dot{R}}{\omega_0}$, and the steady state error for the second-order filter is $\frac{\ddot{R}}{\omega_0^2}$ [29].

Initially the B_{n_F} and B_{n_P} were set at 20 Hz for both the FLL and PLL portions of the filter. This value was then decreased as the PIT increases. This will be described later in Section 3.6.2. The output from this filter is the current change in the Doppler Shift. This is then passed to the NCO to update the carrier timing of the tracking loop and will be further discussed in Section 3.5.6.

3.5.5 DLL Filter.

The DLL filter design for this receiver is a first-order loop. The B_{n_D} is determined using Equation (13). This is the same filter design used for the FLL described in the previous section since both loops are first-order filters. The initial B_{n_D} for the DLL filter was set at 5 Hz. The input to this filter is the DLL discriminator output (in units of chips) and the filter output is in units of chips/second. This filter output is sent to the code time NCO in order to update the speed of the reference PRN code generator.

3.5.6 NCO Updates.

As previously mentioned in Sections 2.5.2.1 and 2.5.2.3, the carrier NCO changes the frequency of the local reference signal while the code NCO changes the rate at which the PRN code is generated. The NCOs rely on inputs from the loop filters described in Sections 3.5.4 and 3.5.5.

3.5.6.1 Carrier Time Updates.

Since the satellite is constantly moving in its orbit, its velocity is changing in reference to the receiver. Additionally, receiver motion (if present) and satellite and receiver clock drifts contribute to an apparent velocity or change in velocity (acceleration). This rate of change of velocity is the change in frequency that must be matched by the NCO to maintain lock on the signal. The output of the FLL/PLL filter is the updated Doppler Shift and then is converted to a value that controls the NCO's carrier time rate change. This value is computed by Equation (17).

$$corr_{carrier} = \frac{f_d}{f_{IF}} \tag{17}$$

where

$$f_d = f_{d_{old}} + \delta f_d \tag{18}$$

f_d is the Doppler Shift (and will be explained further in Section 3.5.7), and f_{IF} is the IF. The $corr_{carrier}$ value is then used in the carrier time updating algorithm to produce the new carrier time vector. Equation (19) illustrates this process.

$$\begin{aligned}\text{[carrier time vector]} = &\ \text{(end of previous carrier time vector)} \\ &+ \text{[1 ms time vector]} \times (1 - corr_{carrier})\end{aligned} \tag{19}$$

where

$$\text{[1 ms time vector]} = \ [T_s\ 2T_s\ 3T_s\ ...\ \text{PIT}]$$

With these vectors formed, the reference signals are created. Equations (20) and (21) represent the locally generated reference signals, I_{ref} and Q_{ref}.

$$I_{ref} = \cos\left(2\pi f[\text{carrier time vector}]\right) \tag{20}$$

$$Q_{ref} = \sin\left(2\pi f[\text{carrier time vector}]\right) \tag{21}$$

Effectively, the $corr_{carrier}$ speeds up or slows down the time vector, which has the effect of increasing or decreasing the frequency of the reference carrier signals.

3.5.6.2 Code Time Updates.

In order for the signal to be tracked, the timing of the PRN code needs to accurate. This requires the code NCO to update the code time so that the locally generated signal's PRN is within the receiver's correlator chip spacing. The output of the DLL discriminator is units of chips which is then sent to the NCO. This receiver also uses velocity aiding from the FLL/PLL tracking loop filter to keep track of the change in the Doppler Shift. Equation (22) shows how the value for the NCO controller is calculated,

$$corr_{code} = \frac{DLL_{out} \times \omega_{0_D}}{R_c} + \frac{f_d}{f_c} \tag{22}$$

where DLL_{out} is the output from the DLL discriminator, ω_{0_D} is calculated from B_{n_D} as described in Section 3.5.5, R_c is the chipping rate of the PRN code, which is 1.023 Mchips/sec, f_d is the Doppler Shift, and f_c is the carrier frequency of the signal, which for L1 is 1575.42 MHz. The $corr_{code}$ is then inserted into Equation (23) in order to update the new timing of the locally generated PRN code.

$$\begin{aligned}[\text{code time vector}] = &\ (\text{end of previous code time vector}) \\ &+ [\text{1 ms time vector}] \times (1 - corr_{code})\end{aligned} \tag{23}$$

3.5.7 Doppler Correction.

The Doppler Shift is continuously being calculated by the receiver from the output of the FLL/PLL tracking loop as previously mentioned in Section 3.5.6.1. This helps maintain the correct frequency for the receiver to use in order to properly track a satellite during its orbit. Figure 3.4 shows the Doppler Shift of PRN 20 for Data

Set 1. It is clearly seen that the Doppler frequency increases throughout the duration of the data set. The other satellites exhibit similar results since each satellites has the same general motion from rising, traveling overhead, and then setting. The Doppler Shifts for each satellite are shown in Figures 3.5 and 3.6 for the first and second data sets, respectively. In GNSS terms, a satellite moving away will have a positive Doppler Shift, while a satellite that is approaching the user will have a negative frequency. A Doppler Shift of 0 Hz indicates that the satellite is directly above the user assuming that the receiver antenna is stationary and the satellite and receiver clock drifts are ignored. However due to the clock drifts, there is rarely a 0 Hz Doppler Shift even with zero relative motion between the satellite and receiver antenna.

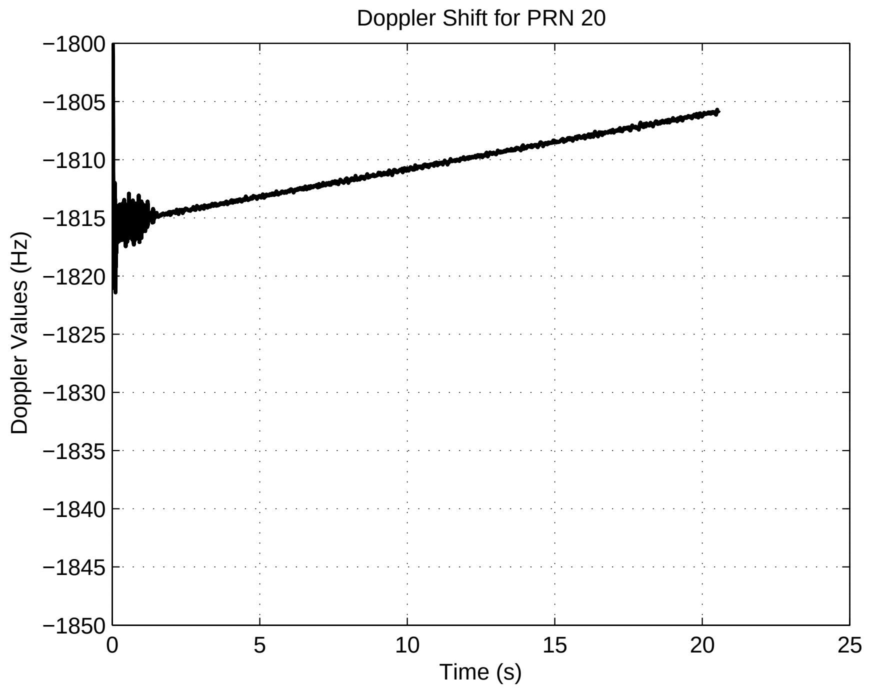


Figure 3.4. GPS Doppler Shift for PRN 20 for Data Set 1. The continual increase of the Doppler frequency is expected due to the path that the satellite travels starting at a rising position, moving overhead of the receiver, and then eventually setting below the horizon.

Also interesting to note is the appearance and disappearance of satellites. The

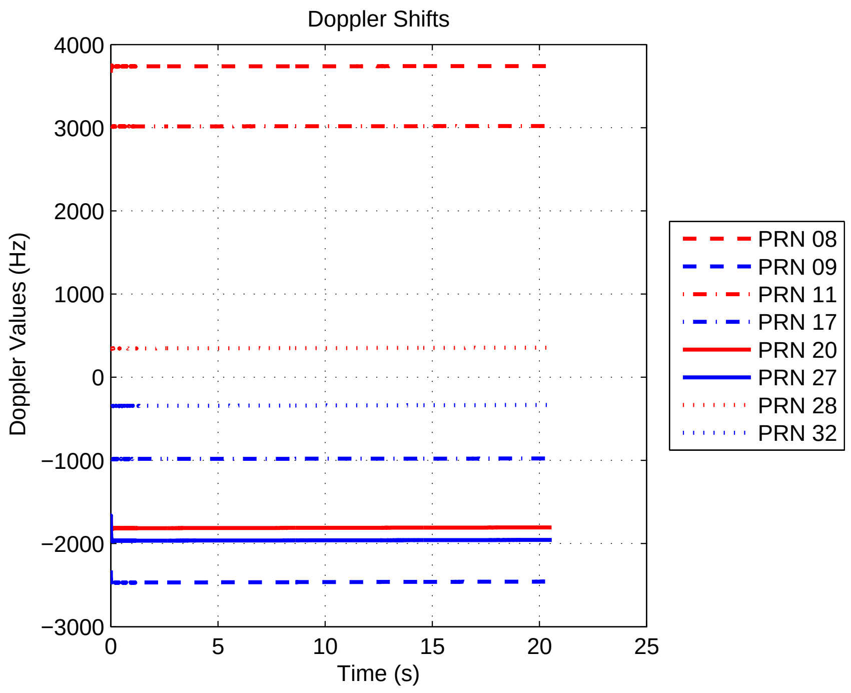


Figure 3.5. GPS Doppler Shifts for Data Set 1

satellites with PRNs 08 and 11 have large positive Doppler Shifts in Data Set 1, indicating that they are about to go below the horizon and out of view of the antenna. As expected in Data Set 2, these satellites are no longer being tracked. Also, the satellite with PRN 02 appeared in the second data set (with a large negative Doppler indicating it just rose above the horizon).

3.6 Post-Lock Calculations

Prior to this point, the receiver was running for a set amount of time in order to determine if the signal lock was present. For this receiver, the signal was determined to be locked on if there was bit synchronization. The following sections will describe bit synchronization as well as other algorithms performed after this synchronization has occurred.

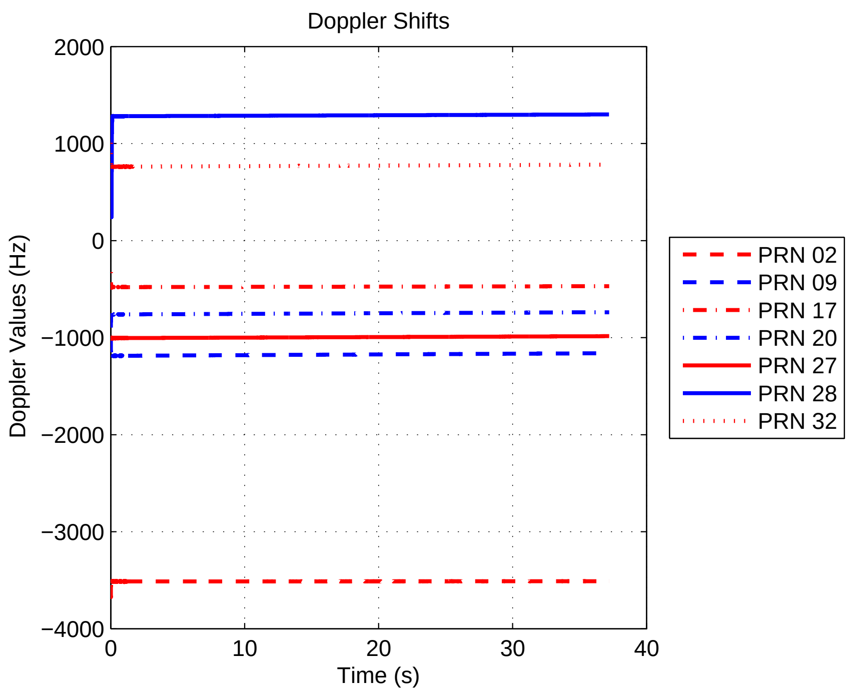


Figure 3.6. GPS Doppler Shifts for Data Set 2

3.6.1 Bit Synchronization Approach.

In order to increase the PIT, decrease the noise bandwidths of the tracking filters and determine the data bits, bit synchronization must occur. The method chosen for bit synchronization was adapted from the approach described in [27]. First the tracking loop is run at 1 ms PIT since there can be a bit flip at any epoch since the data bit boundaries are unknown to this point. Bit flips are then counted by the receiver and stored into location bins. Since the L1 C/A code data message contains 20 ms bits, there are 20 bins possible for a bit flip. A bit flip is determined by comparing the sign of the previous epoch to the current epoch. Nominally, if a signal is being tracked properly, one bin will begin to accumulate most of the bit flips. Therefore, a count threshold and bit ratio were chosen to ensure proper data bit locations.

For this receiver, the count threshold was set to 15 and the bit ratio was set at 3.

These values were determined by trial and error in order to filter out signals that could not be tracked due to low C/N_0, but still allow stronger signals to successfully be tracked. After one second of data is processed, the bins are sorted from highest to lowest. If the first bin surpasses the count threshold, then the first bin is divided by the second bin and compared to the bin ratio. If the bin ratio is surpassed, then bit synchronization has been accomplished. The first bin's corresponding millisecond location is determined to be the start of a data bit. Figure 3.7 shows an example of a successful bit synchronization bin histogram. Figure 3.8 is an example of bit synchronization failure.

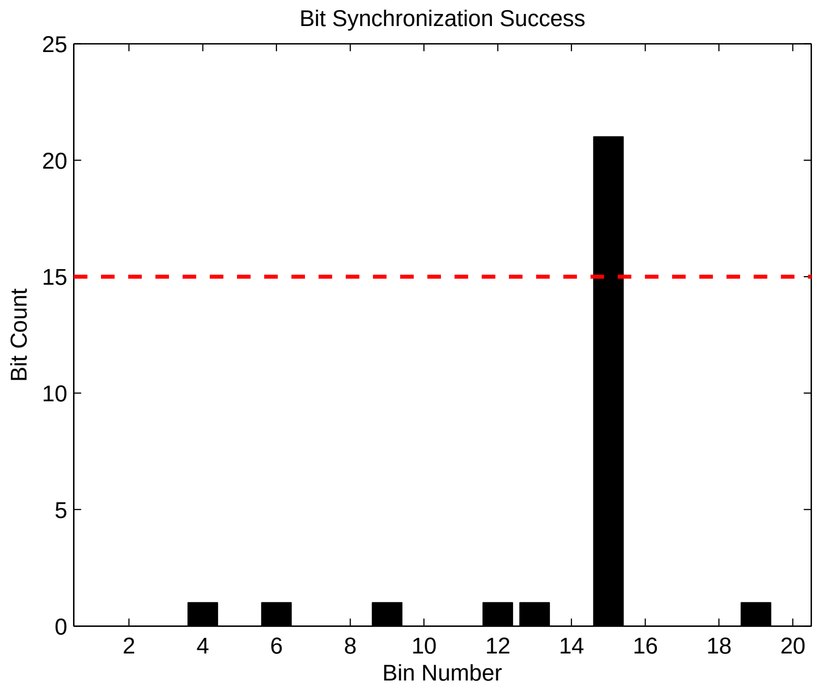


Figure 3.7. Histogram of a successful bit synchronization. As can be clearly seen, the number of data bit transitions exceeds the threshold of 15, and the bit ratio is greater than 3.

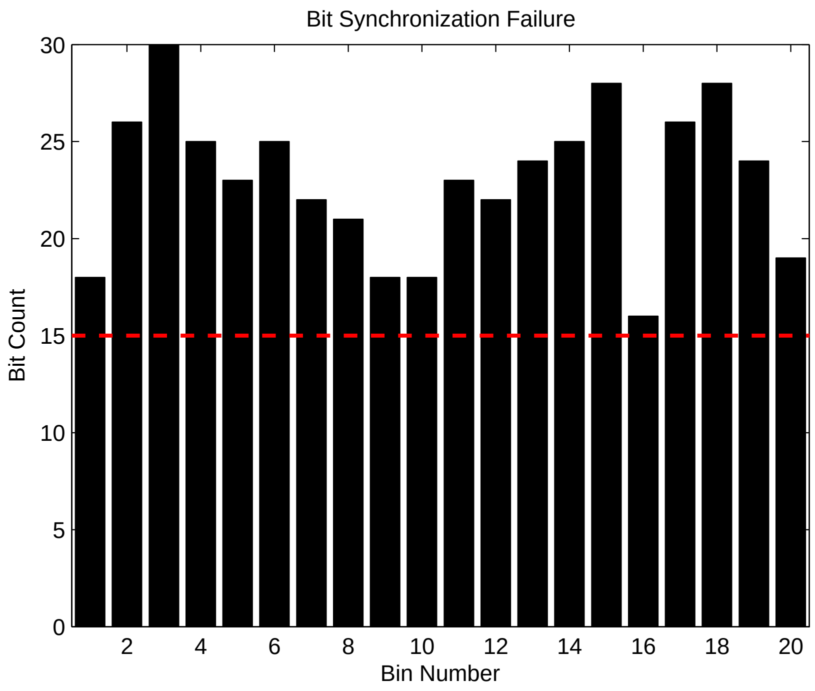


Figure 3.8. Histogram of an unsuccessful bit synchronization. The threshold was set to 15 and the bit ratio was set to 3. Numerous bins exceed the threshold, indicating that a lock on the signal did not occur.

3.6.2 PIT & B_n Changes.

Once the location of data bit transitions is determined, the PIT is increased and the B_n is decreased to provide better tracking capabilities, because discriminator outputs are less noisy with longer PITs. In reality, the receiver is always integrating the signal with the reference copy at 1 ms intervals, but the PIT is changed by summing up the resulting integrations for different amounts of time. The receiver is configured to increase PITs in steps in order to ensure continued lock of the signal. After the data bit start location is determined from the bit synchronization algorithm, the PIT is then increased to 2 ms for a length of 200 ms. One must note that the length of time that a PIT is running must be a multiple of 20 ms in order not to have a possible data bit transition during the PIT. The B_{n_P} is decreased to 15 Hz while the B_{n_D} is decreased to 1 Hz. After the 200 ms, the PIT is then increased

Table 3.1. Summary of PIT and B_n changes. Note that B_{n_f} is not included because the FLL is turned off after 120 ms of data is processed.

Time (s)	*PIT (ms)*	B_{n_P} *(Hz)*	B_{n_D} *(Hz)*
0	1	20	5
1	2	15	1
1.2	4	12	0.25
1.4	20	10	0.05

to 4 ms for another 200 ms, and B_{n_P} is decreased to 12 Hz and B_{n_D} is lowered to 0.25 Hz. Once again after 200 ms, the PIT is increased to its final length of 20 ms, and B_{n_P} is decreased to its final setting of 10 Hz and B_{n_D} is lowered to 0.05 Hz. These values were chosen based on trial and error, and after each decrease the discriminator outputs were stable after each 200 ms step. Table 3.1 summarizes the changes in PITs and B_ns at the appropriate times for this receiver.

Figure 3.9, Figure 3.10 and Figure 3.11 show the FLL, PLL and DLL discriminator outputs, respectively, after the first 2 s of a data collection to clearly show the reduction in noise from the outputs as the PIT and B_n changes. Figure 3.12 and Figure 3.13 show the PLL and DLL discriminator outputs, respectively, for a full data run. As stated earlier, the FLL is turned off after 120 ms and the PLL does not start until 40 ms. Therefore, the full FLL discriminator plot would only display 0 after 120 ms and is not included. Figure 3.14 displays the time domain scatter plot which highlights the change in magnitude in the correlator normalized values as the PIT increases. As is clearly seen, most of the signal is located in the In-phase portion of the signal.

3.6.3 Carrier-to-Noise Density Estimation Calculation.

Estimating the Carrier-to-Noise Density (C/N_0) provides an estimate of the receiver signal strength. This method is taken from [27]. C/N_0 is estimated by compar-

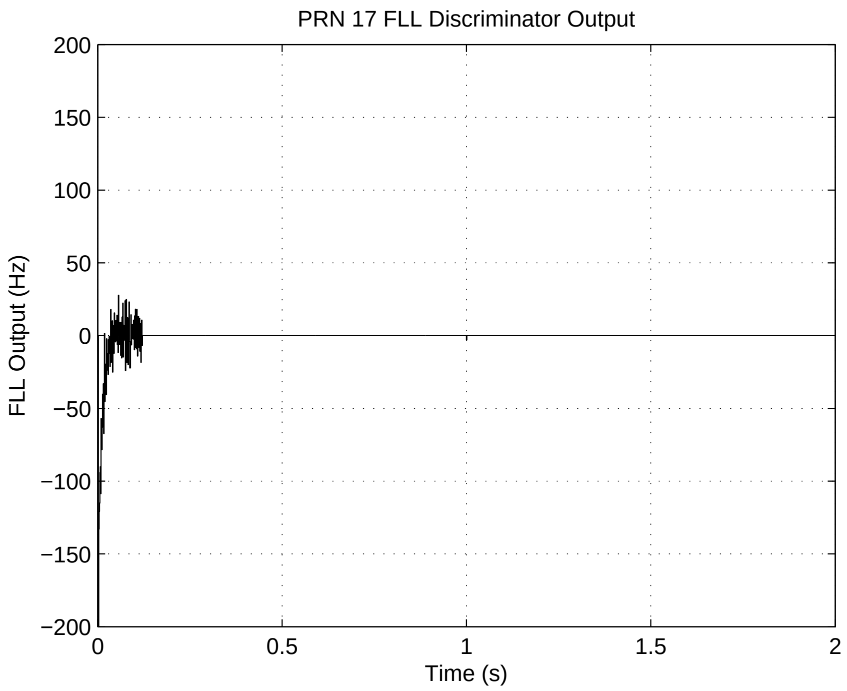


Figure 3.9. PRN 17 FLL Discriminator Output Over 2 Seconds. Note that the FLL is turned off after 120 ms of processing data.

ing signal-plus-noise power in two different bandwidths. Total power measurements are determined in $1/T$ noise bandwidths (wide-band power–WBP) and $1/MT$ noise bandwidths, (narrow-band power–NBP). Equation (24) details how WBP was calculated and Equation (25) shows how NBP was determined.

$$WBP_k = \left(\sum_{i=1}^{M} (I_{P_i}^2 + Q_{P_i}^2) \right)_k \tag{24}$$

$$NBP_k = \left(\sum_{i=1}^{M} I_{P_i} \right)_k^2 + \left(\sum_{i=1}^{M} Q_{P_i} \right)_k^2 \tag{25}$$

Each k^{th} value is the summation of the past M samples of prompt In-phase (I_P) and Quadrature (Q_P) correlator outputs. These outputs are taken at 1 ms intervals before the summation of correlator outputs if the PIT is longer than 1 ms. For this

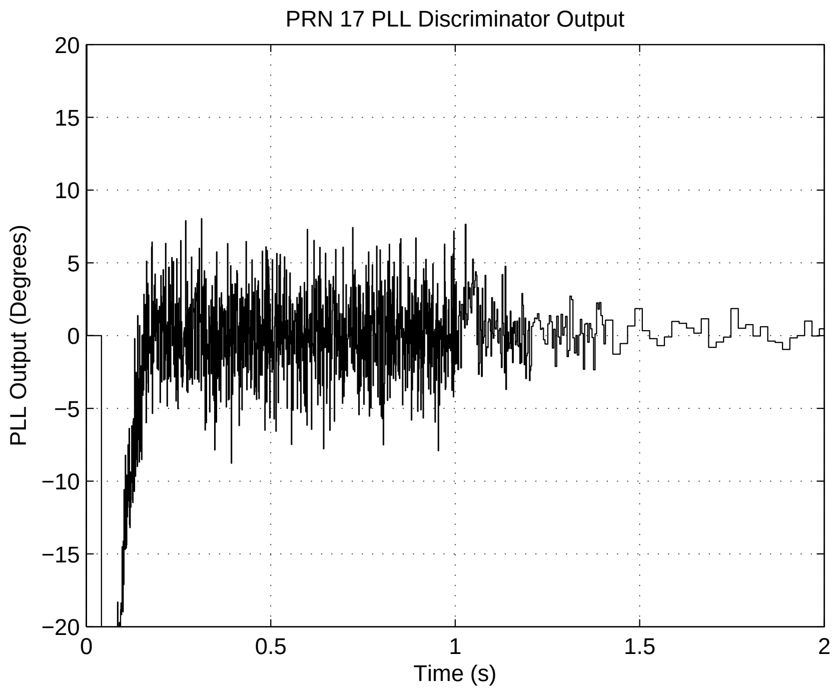


Figure 3.10. PRN 17 PLL Discriminator Output Over 2 Seconds. Note the decrease in the noise at each PIT & B_n change interval.

receiver, M was set to 20 samples. Once these values are determined, a normalized power (NP) was defined as

$$NP_k = \frac{NBP_k}{WBP_k} \tag{26}$$

Once NP is calculated, there are two methods described in [27] to determine the C/N_0. The first method is based off of statistics from NP. The mean and standard deviation of NP can be used to determine the C/N_0 by looking at Figure 3.15 based on a $M = 20$.

For the second method after NP is calculated, Equation (27) determines the mean normalized power ($\hat{\mu}_{NP}$) after K measurements. This receiver has K set to 50.

$$\hat{\mu}_{NP} = \frac{1}{K}\sum_{k=1}^{K} NP_k \tag{27}$$

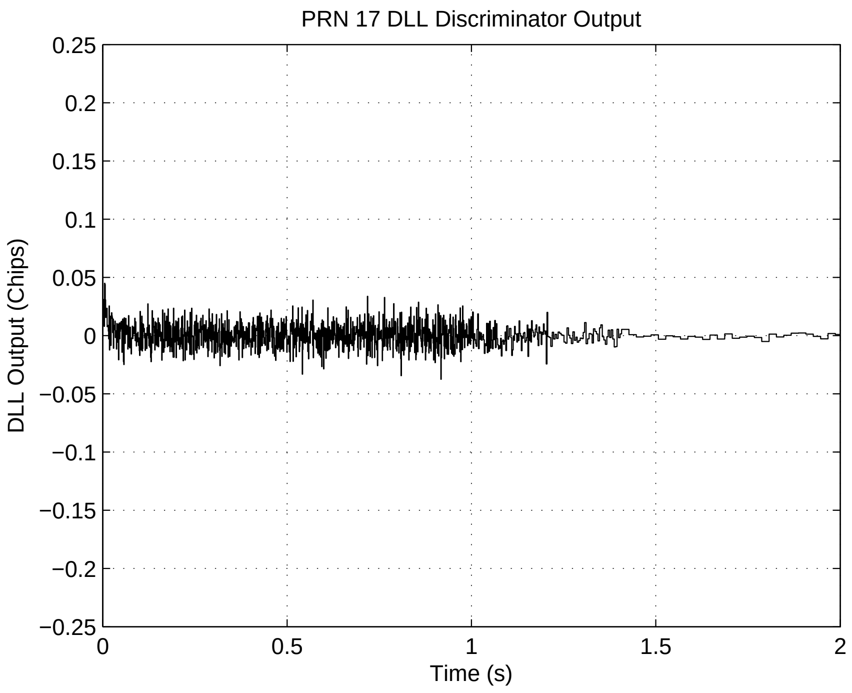


Figure 3.11. PRN 17 DLL Discriminator Output Over 2 Seconds. Note the decrease in the noise at each PIT & B_n change interval.

This measurement is then used in the following C/N_0 estimator described as

$$\frac{\hat{C}}{N_0} = 10\log_{10}\left(\frac{1}{T}\frac{\hat{\mu}_{NP}-1}{M-\hat{\mu}_{NP}}\right) \tag{28}$$

C/N_0 measurements are then updates every second since $M = 20$ and $K = 50$. Figure 3.16 shows the C/N_0 estimates for Data Set 1. Figure 3.17 shows the C/N_0 estimates for Data Set 2. These values are very reasonable for GPS power levels. Ohio University also calculated some C/N_0 values from Data Set 2 with their software receiver in order to ensure that the AFIT in-house software receiver was working properly. These values are presented in Figure 3.18 [14]. As is clearly seen, the AFIT and Ohio University values are very similar, validating the AFIT software receiver. Although the software receivers differ, it must be noted that all values by both receivers are from the C/N_0 estimation approach given by Equation (28). The

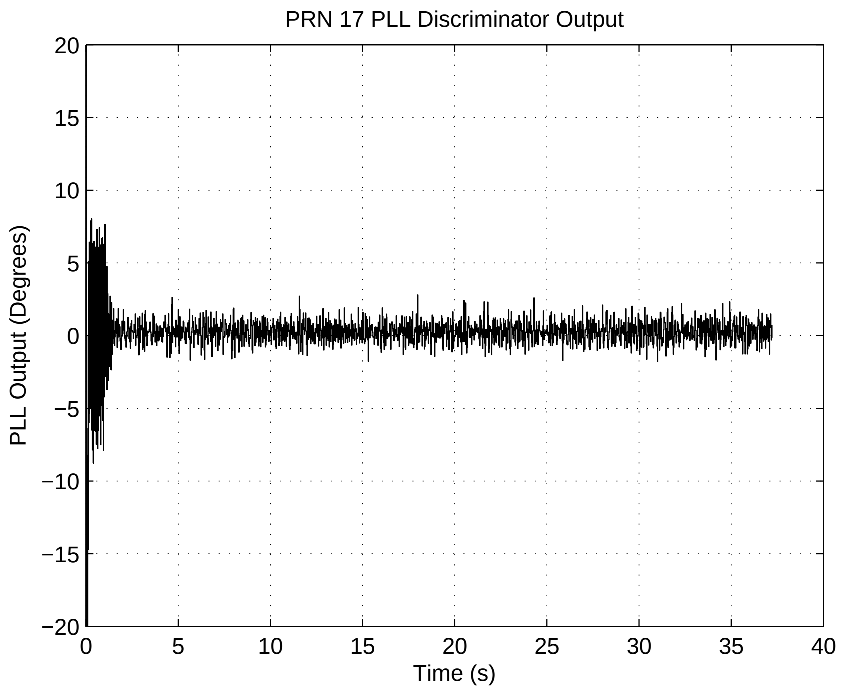


Figure 3.12. PRN 17 PLL Discriminator Output for Data Set 2.

satellites in Figure 3.18 are the only satellites that were both tracked by the AFIT and Ohio University receivers.

3.7 Post-Processing Results

After the acquisition and tracking portions of receiver have been accomplished, the final output of these is the data message. Figure 3.19 shows the entire message from the full data run, and Figure 3.20 shows a shorter portion to highlight the data bits. Once the data message was output, the decoding of the message began.

3.7.1 Data Bit Determination & Decoding.

The output message was composed of 1s and -1s and was mapped to 0s and 1s respectively for the decoding process. With the structure of the GPS data message as defined in the ICD-GPS-200c [1] known, the data message is easily decoded. A

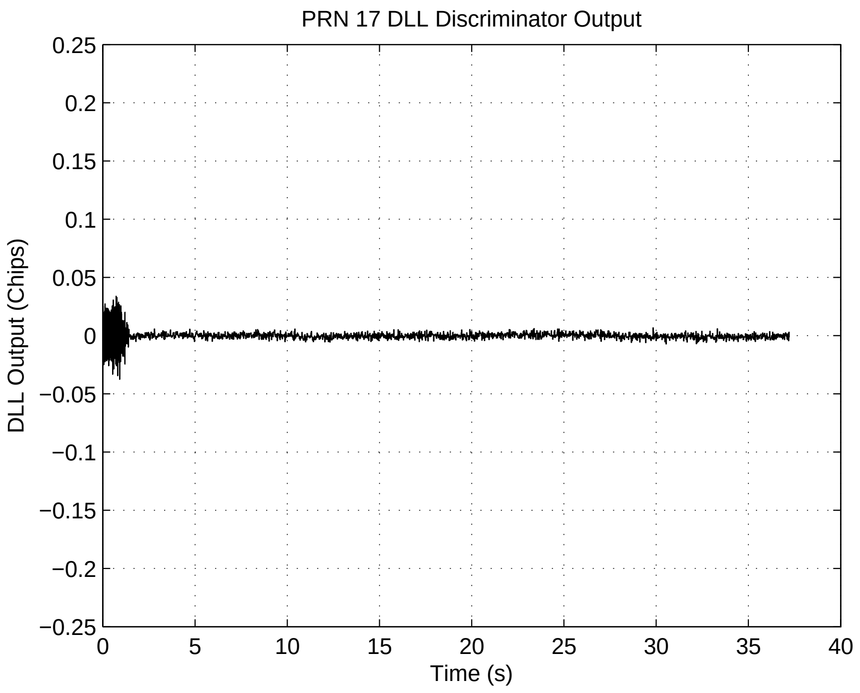


Figure 3.13. PRN 17 FLL Discriminator Output for Data Set 2.

complete frame from the GPS data message is 30 s long and composed of five 6 s subframes each containing 300 bits. Each subframe begins with a defined preamble consisting of 8 bits: 1 0 0 0 1 0 1 1. First, the data bits are correlated with the preamble to determine possible correct locations of the beginning of a subframe. A check for the preamble with sign ambiguity is conducted in case there is a sign change on the message since the receiver can really only track bit changes, not absolute bit values. It must be noted that the preamble can sometimes be found at random points in the message, since the data bits can contain the same combination of bits. Therefore, once location of the preamble are located, an algorithm is run to determine that the preambles are spaced about by 300 bits (the length of a subframe). Once the start of subframes are located, the message components can be deciphered. The next portion of the message that is meaningful to this receiver is the Handover Words (HOW) because it contains the subframe identifier and Z-count, which will be described in

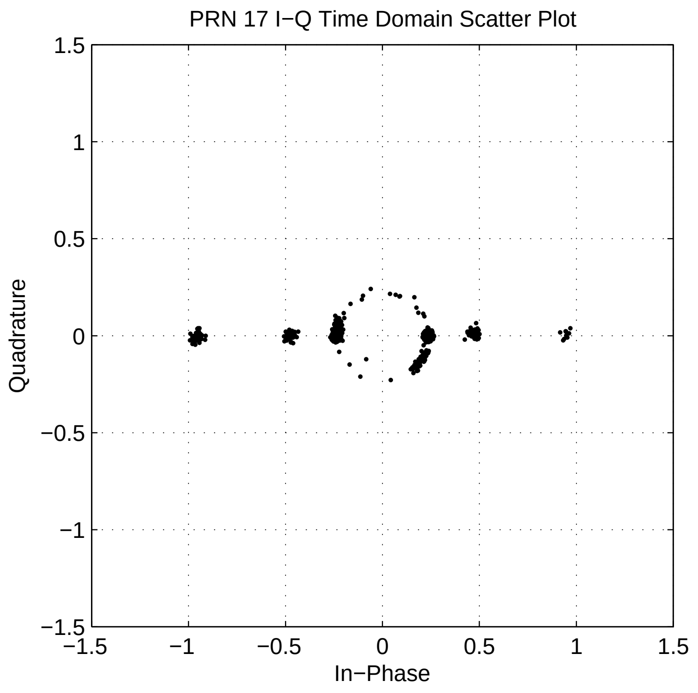


Figure 3.14. PRN 17 Time Domain Scatter Plot. This illustrates that all the data message is in the In-Phase portion of the signal. The tight groupings indicate a strong lock on the signal. Note that the points start near the center and then increase due to the increasing PIT. The points that are off of the I-axis occurred during initial pull-in.

the following section.

3.7.2 HOW Decode & Z-Count Determination.

The purpose of this receiver is to determine pseudoranges in order to calculate position. Therefore, only the Time of Transmission (TOT) was needed to determine precise timings for the GPS signal. The TOT is interpreted from data in the HOW. The first 17 bits of the HOW are the truncated Z-count. The Z-count is a Time of Week (TOW) counter that increments every 1.5 seconds. Therefore, to determine the true TOW, the Z-count must be multiplied by 1.5. This TOW is the TOT for the next subframe, but in order to determine the TOT for the current subframe, 6 s must be subtracted from the TOT. This TOT then becomes a reference time for the

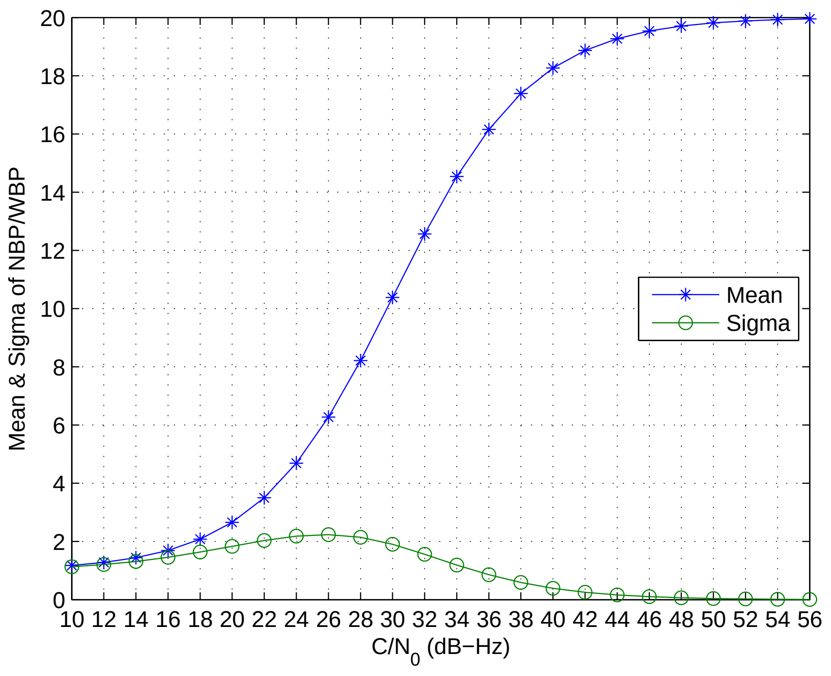


Figure 3.15. Statistics of NP for $M = 20$.[27]

pseudorange to be calculated.

3.7.3 Pseudorange Calculation.

After the TOT is determined for a specific signal, the satellite's transmit time can be estimated. Since there are several satellites, the transmit time must synchronized for all of the satellites. This was accomplished by calculating a millisecond offset value, which is the difference between the TOT and the receiver's reference time at the moment of transmission. This value is then rounded to the nearest millisecond and added to the reference clock. This produces the transmit time of the signal, excluding clock errors.

Next the received time was calculated. First a sample travel time was estimated by subtracting the first transmit time value from the first received time value. Then a desired travel time was calculated a desired pseudorange value of 20,000 km to

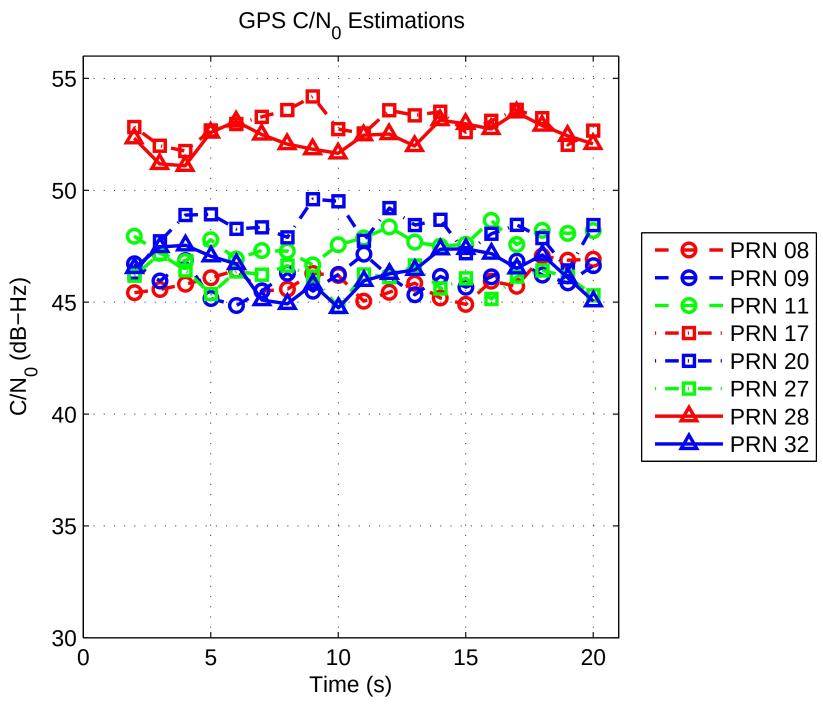


Figure 3.16. C/N_0 **Estimates of GPS Satellites for Data Set 1.**

estimate an average signal travel time. Finally, the receiver time correction was calculated by subtracting the desired time from the sample time. This correction value was then subtracted from the actual receiver time to produce the received time, excluding clock errors.

After all the timing corrections were made, the actual pseudoranges were estimated. Equation (29) represents the pseudorange estimate calculation,

$$\rho = c(T_r - T_t) \tag{29}$$

where ρ is the pseudorange, c is the speed of light, T_r is the receiver time, and T_t is the transmit time. Since the TOT is only transmitted every subframe, the pseudoranges must be interpolated to fill in the time between the 6 s gaps. The *interp1* MATLAB function was used for this task and created all of the pseudorange estimates.

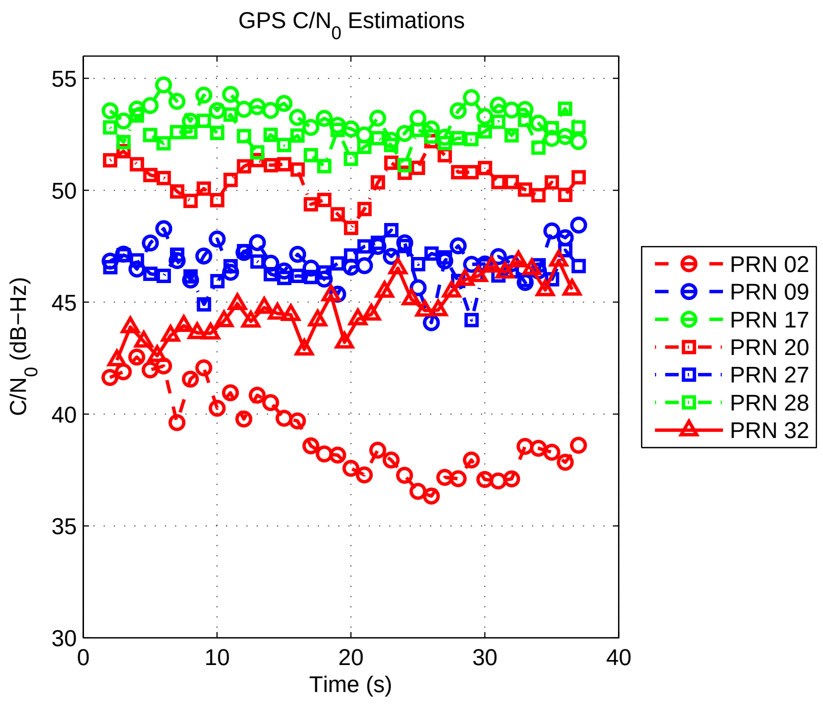


Figure 3.17. C/N_0 **Estimates of GPS Satellites for Data Set 2.**

3.7.4 Position Estimate and Errors.

Once all the pseudoranges were calculated, the ephemeris for the satellites was obtained to begin user position estimation. With the ephemeris from the day of data collection, the satellites' orbits can be determined. Next the actual range of the satellites are used in an iterative least squares process similar to that shown in [21] to calculate receiver (antenna) position and clock error. Then the true position of the antenna is compared to the estimated position in order to calculate position error. Figure 3.21 shows the errors in the North, East and Vertical directions for Data Set 1, and Figure 3.22 shows the errors in the North, East and Vertical directions for Data Set 2.

The vertical measurement typically is the largest position error from a receiver due to the largest pseudorange uncertainty being in the tangential vertical direction. Therefore if the vertical error is ignored, a surface error can be estimated from the re-

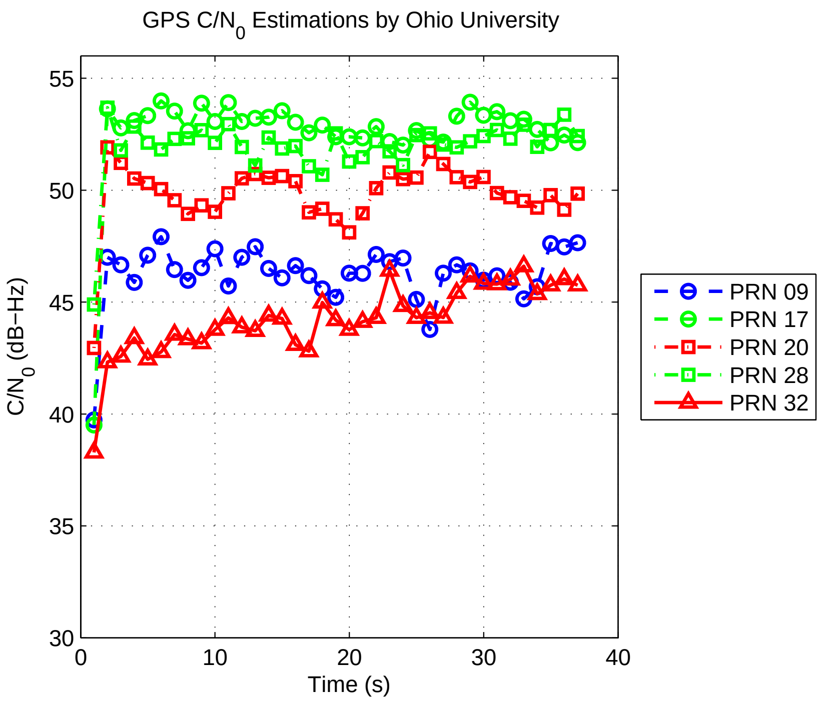


Figure 3.18. C/N_0 **Estimates of GPS Satellites for Data Set 2 by Ohio University. Note the excellent agreement from the different receivers. The same** C/N_0 **estimation from [27] was used by both receivers. [14]**

maining North and East direction errors. Figures 3.23 and 3.23 show the North versus East directional errors from Data Set 1 and Data Set 2, respectively. The respective residual errors from the position estimates are presented in Figures 3.25 and 3.26.

The position errors from the first data set are significantly better than the position errors of the second data set. Although there are many factors that can cause errors in positioning, one important factor here is the fact that the first data set had 8 satellites to estimate a position while the latter data set only had 7 satellites. While only 4 satellites are needed to determine position and clock error, the more satellites available will allow a more precise position estimation. Many commercial GPS receivers provide errors under 10 meters. However, since this receiver did not incorporate any error mitigation algorithms for such errors caused by the ionosphere and troposphere, obtaining a position error of under 10 meters validates the receivers

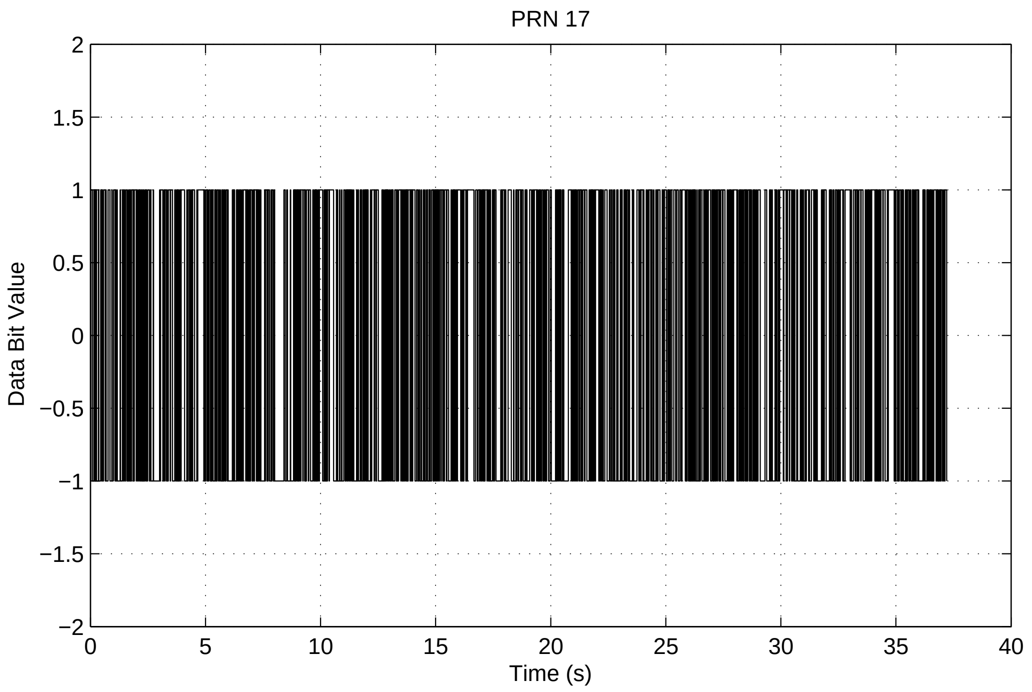


Figure 3.19. PRN 16's Data Message

capability to acquire, track and provide a reasonable position.

3.8 Summary

Chapter 3 discussed the methods used by the software to acquire and track GPS satellites. First the correct PRNs were needed. Once these were generated the satellite needed to be acquired and then successfully tracked by the receiver. Proper tracking allowed the decoded of the navigation message to allow pseudorange estimates. Once all the pseudoranges were calculated, the user's position was estimated and was shown to be reasonable, given that no additional error mitigation techniques were applied.

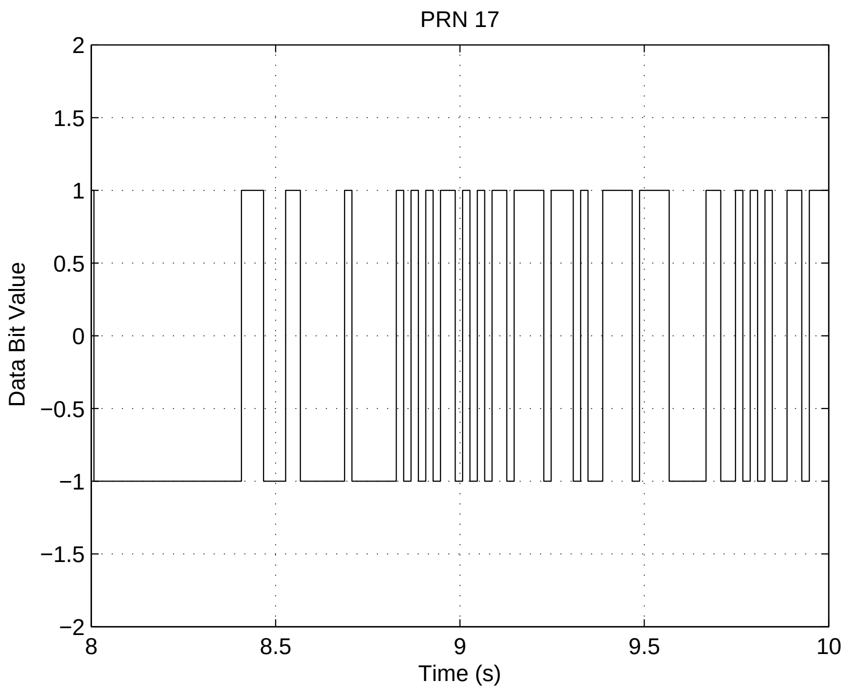


Figure 3.20. PRN 17's Data Message Expanded View to Highlight Data Bits

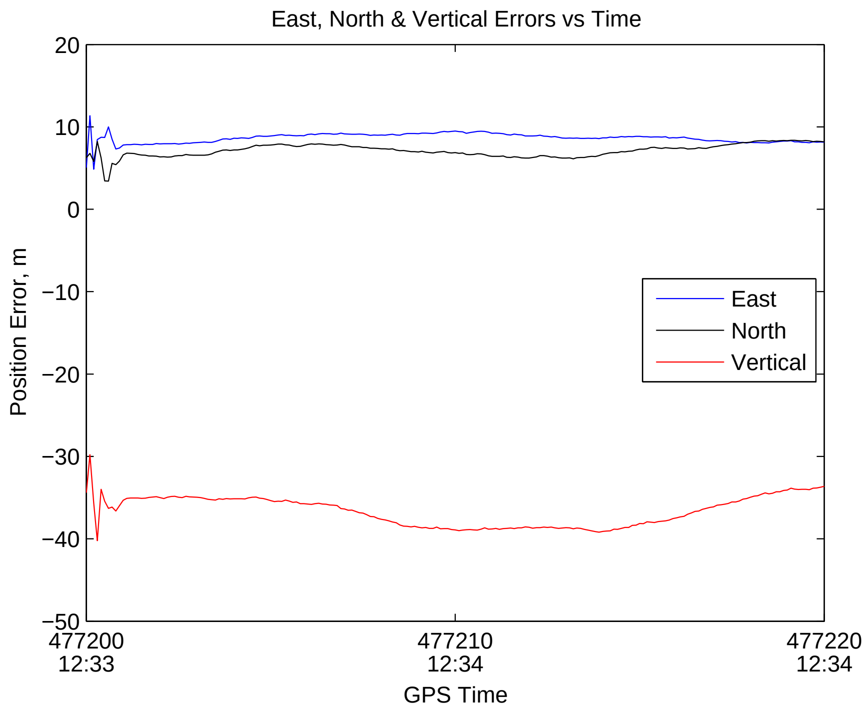


Figure 3.21. Position Error: North, East & Vertical for Data Set 1.

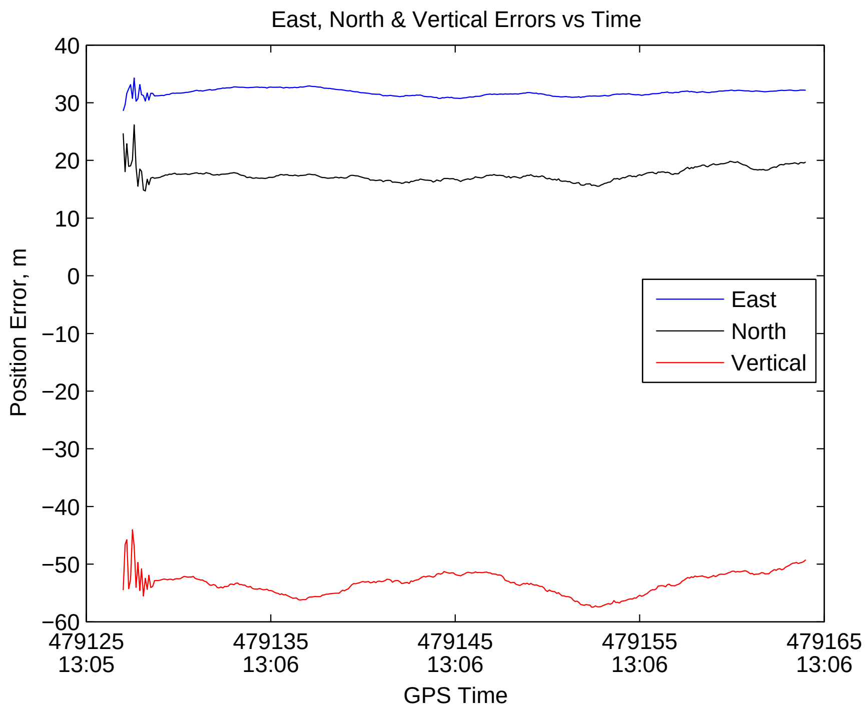


Figure 3.22. Position Error: North, East & Vertical for Data Set 2.

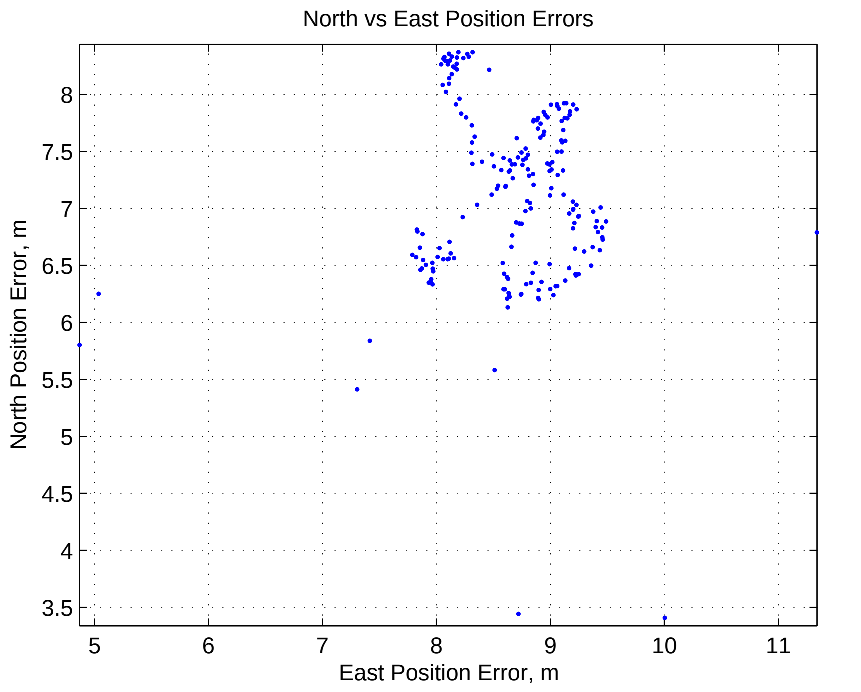


Figure 3.23. Position Error: North & East for Data Set 1.

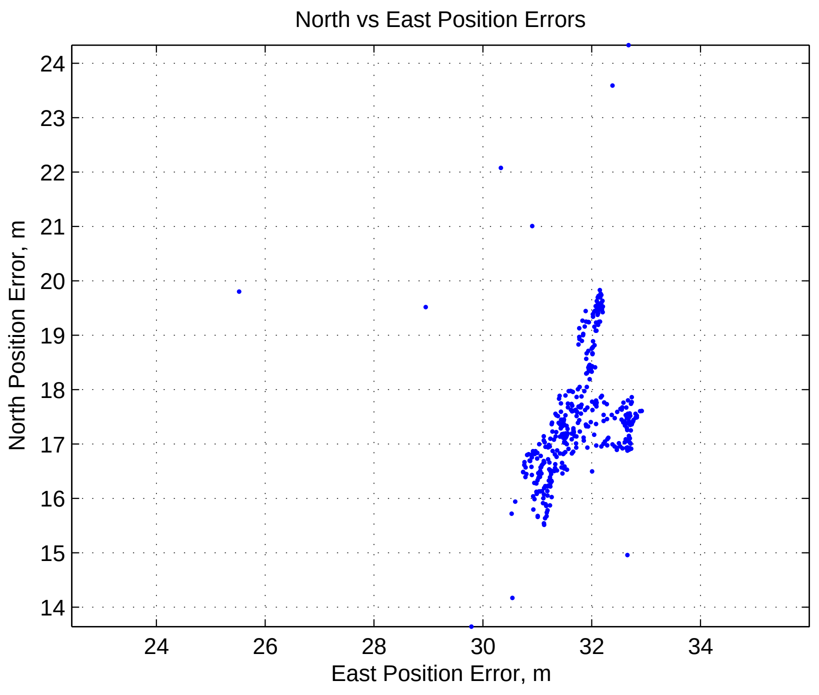


Figure 3.24. Position Error: North & East for Data Set 2.

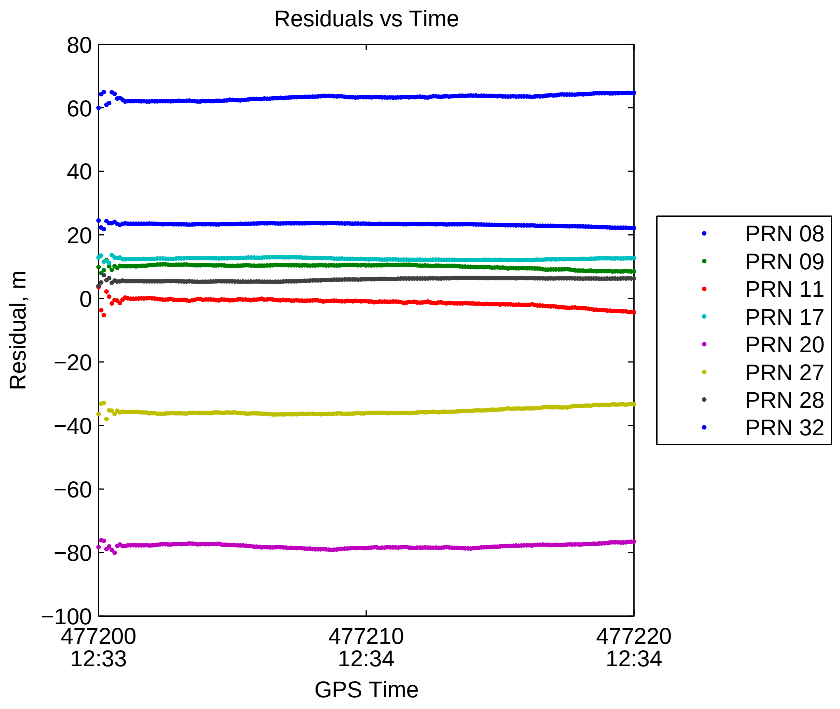


Figure 3.25. Position Residuals for Data Set 1

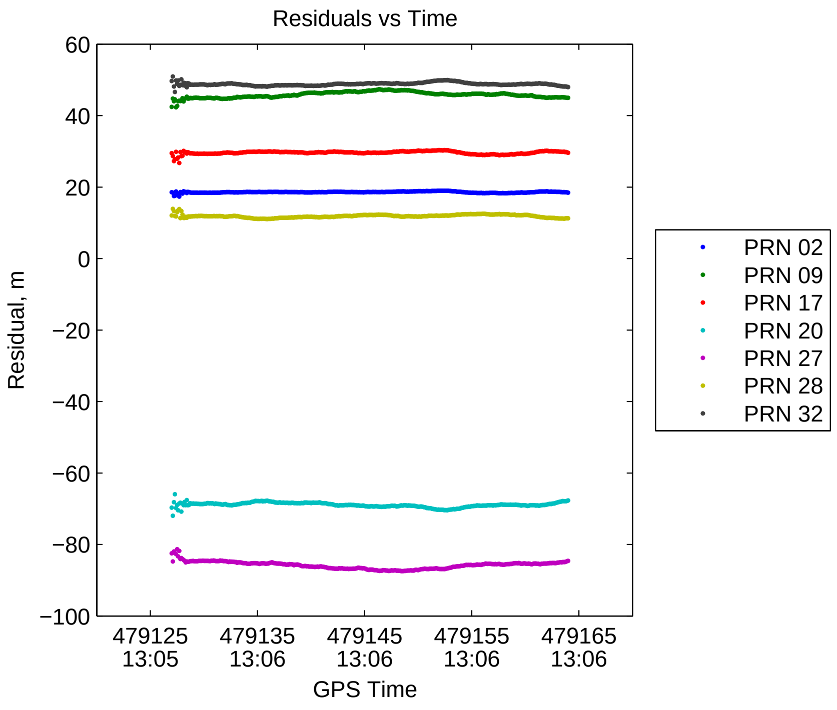


Figure 3.26. Position Residuals for Data Set 2

IV. Compass and Galileo Software Defined Radio Modifications

4.1 Overview

This chapter describes the modifications made to the GPS SDR described in the previous chapter to track both Compass and Galileo signals. All the data collected for this GNSS SDR was collected in the E2-L1 bands from Figure 2.1. Therefore, only the Compass E2 and Galileo L1 signals are used in this receiver (in addition to GPS L1). The receiver has different configurations dependent on which satellite system is currently being tracked. The SDR parameter file is configured to load certain parameters for each GNSS. Since Compass and Galileo do not have full constellations in orbit, pseudorange and position estimates were not computed (a pseudorange for Compass cannot be determined since there is no official documentation of the signal structure at this time). The remaining results will be presented following the respective GNSS SDR modifications for each system. Section 4.2 details the modifications made to track Compass, and Section 4.3 explains the modifications made to track Galileo.

4.2 Compass GNSS SDR Modifications

The following sections detail the appropriate changes made under this research to incorporate the successful acquisition and tracking of the Compass E2 signal. The Compass signal structure is different from that of GPS and Galileo due to the inclusion of a secondary code mixed with the primary PRN. After the secondary code is applied properly, the rest of the receiver works like the GPS portion except for some initial parameter changes.

Table 4.1. Compass E2 I-Channel 11-Stage Gold Code Polynomials and Initial States[9]

Polynomial 1	$X^{11} + X^{10} + X^9 + X^8 + X^7 + X + 1$
Initial State 1	[0 1 0 1 0 1 0 1 0 1 0]
Polynomial 2	$X^{11} + X^9 + X^8 + X^5 + X^4 + X^3 + X^2 + X + 1$
Initial State 2	[0 0 0 0 0 0 0 1 1 1 1]

4.2.1 Compass PRN Generation.

In order to generate the Compass E2 PRN, the polynomials for the 11-stage Gold code must be known. The previous work by Stanford University [9] provided these expressions, which are listed in Table 4.1. A MATLAB script was written to generate the 2046 chip PRN with these given expressions.

4.2.2 Compass Signal Acquisition.

Compass signal acquisition is similar to GPS signal acquisition. As mentioned in the previous section, the E2 signal has a PRN code consisting of 2046 chips with a 2.046 Mchips/sec chipping rate (so it repeats every 1 ms). Although the E2 signal has a 20 bit secondary code, it is not needed to account for this in the acquisition process since the secondary code changes only between 1 ms integration intervals. Since the primary code's chipping rate is twice that of GPS, more bandwidth is required for receiving the signal. Therefore, the averaging method used in the GPS case did not work for Compass (note that further research must be accomplished to determine the proper number of samples needed to be averaged). The use of the non-averaging method entails a longer acquisition but also provides a better resolution of the code phase since all 56,320 samples are used per millisecond block as apposed to the 5120 samples per block used for GPS. The same FFT search algorithm was used to calculate the Doppler Shift and precise code phase location. Also, the same peak matching algorithm was used to determine if consecutive millisecond blocks contained a peak at the same location ensuring a successful acquisition. Figure 4.1 shows the peak of the Compass signal from Data Set 1.

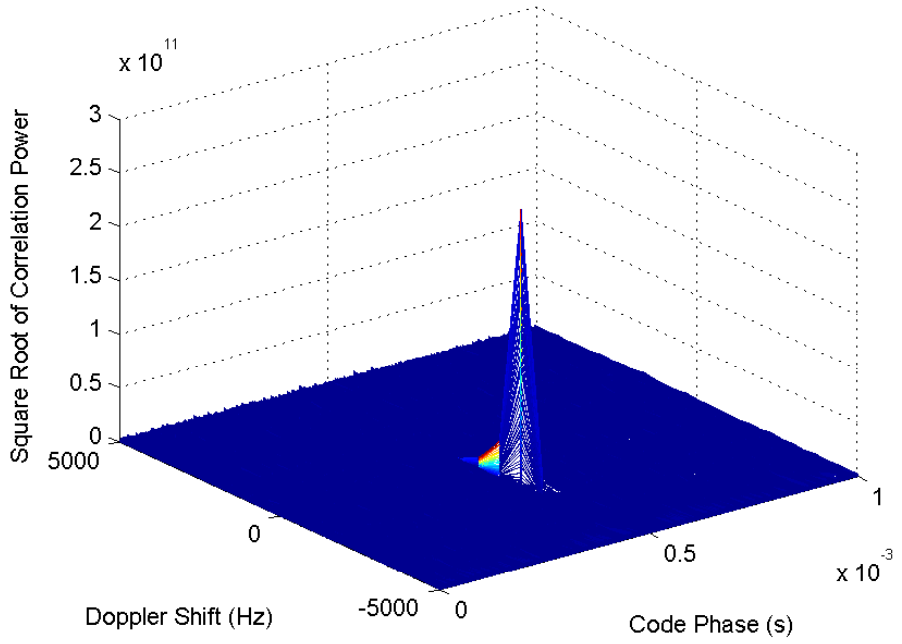


Figure 4.1. Successful Acquisition of Compass Signal

4.2.3 Compass Secondary Code & Tracking.

The Compass data bit structure is the same as GPS, since a data bit is 20 ms in length. However due to the secondary code, each millisecond is multiplied by one of the secondary code's bits. Therefore, in order to successfully track the Compass E2 signal, the secondary code must be determined and then removed from the signal. A previous secondary code: -1 -1 -1 -1 -1 1 -1 -1 1 1 -1 1 -1 1 -1 -1 1 1 1 -1 had been published by a group from Stanford University [9]. This secondary code worked correctly for the first data set.

Unfortunately, this code did not correctly line up with the current broadcast secondary code on the second data set. In order to determine the new secondary code, a 400 ms section was processed by the GPS tracking loop. Since the only difference between tracking GPS and Compass is the secondary code, the tracking algorithm output the Compass signal with the primary code removed. Figure 4.2 shows a 60 ms portion from the output signal of the 400 ms run. Looking at this output, the new secondary code can be seen, since it repeats every 20 ms. The newly determined Neuman Hoffman secondary code for the Compass E2 signal is -1 -1 -1 -1 -1 1 1 -1 1 1 -1 1 -1 1 -1 -1 -1 1 1 -1. Comparing the previously published secondary code to the newly determined code, the codes only differ by two bits. Bits 7 and 17 have been flipped in the new code. It is unknown if there is a predetermined structure to the changing secondary codes, but it is worth investigating in the future.

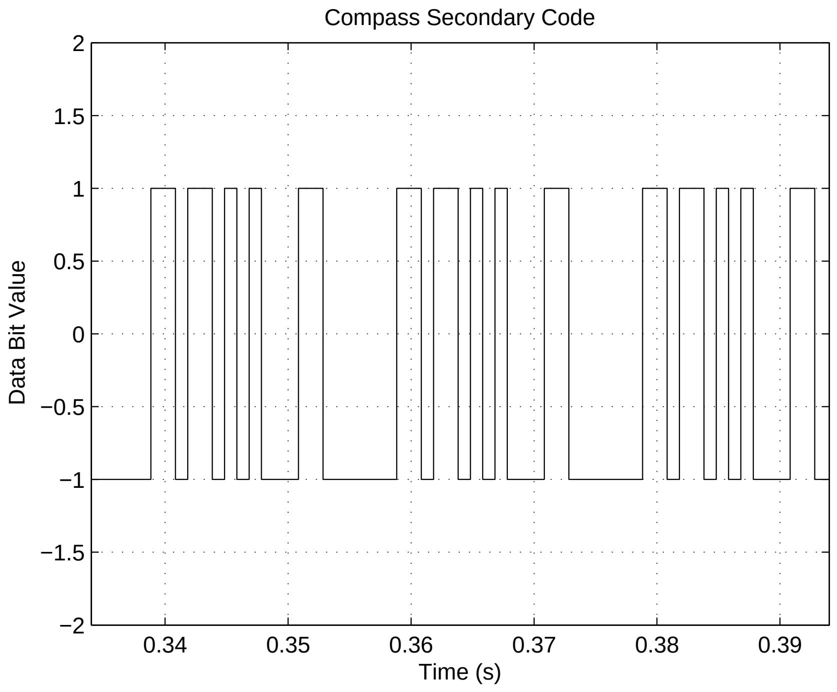


Figure 4.2. 60 ms interval of Compass signal highlighting secondary Neuman Hoffman code. Note the 20 ms secondary code repeating intervals.

Next, the starting bit of the secondary code is determined by rerunning the tracking loop with the secondary code until the output is 20 ms long bits. After the correct start bit of the secondary code is determined, the tracking process for Compass is the same for GPS. However, some of the receiver parameters–bit synchronization and B_n values–were changed in order to better track the Compass signal. These parameter changes will be further discussed in Sections 4.2.4 and 4.2.5, respectively. The receiver tracked the Compass E2 signal while holding the PIT at 1 ms in order to determine that tracking was successful. Figure 4.3 is the full processed data output from the first data set, and Figure 4.4 shows a smaller section of this output to highlight the data bits. Similarly, Figure 4.5 is the full processed data output from the second data set. Clearly shown in Figures 4.3 and 4.5, are large gaps in data bits. It was stated in previous research that navigation data is not always transmitted by the Compass satellite [9]. These gaps are approximately 5 s long with a 1.2 s length burst of data. The gaps actually were beneficial in determining the secondary code since there were no data bit flips to hinder the decoding.

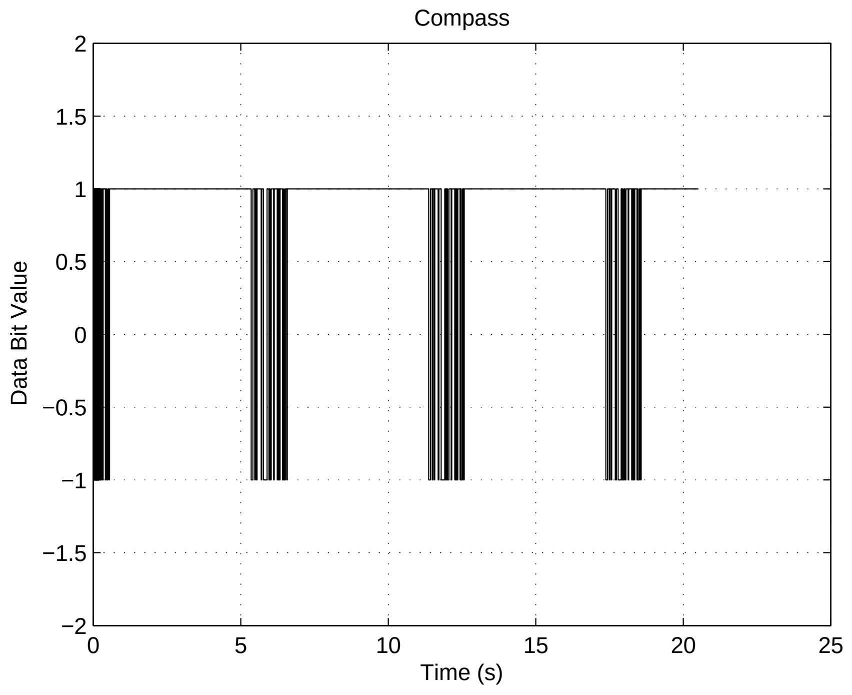


Figure 4.3. Compass' Data Message for Data Set 1. Note the 6 s cycle with approximately a 1.2 s data burst followed by 5 s of no data.

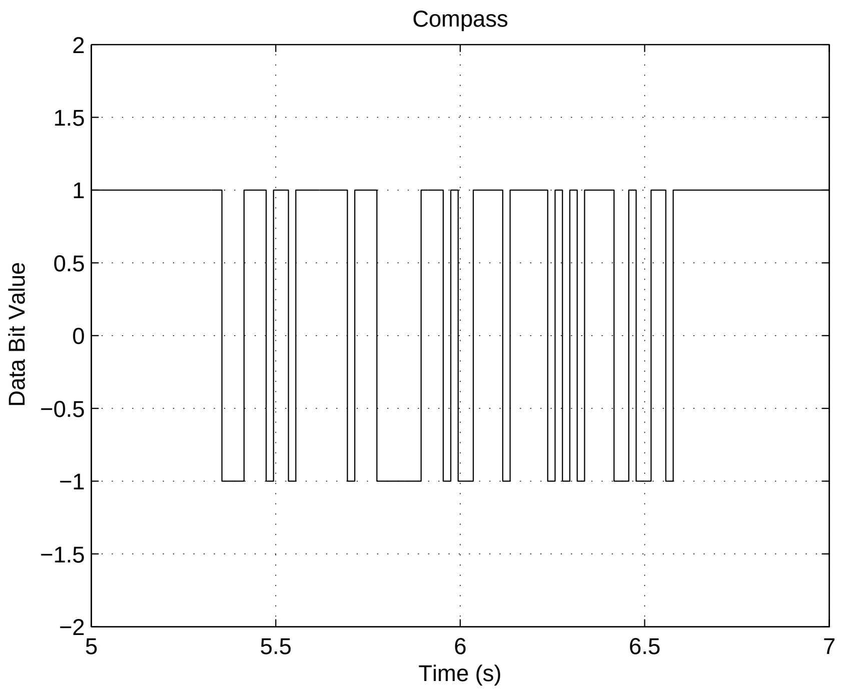


Figure 4.4. Compass' Data Message for Data Set 1 Expanded to Highlight Data Bits

The lock on the signal appears to be lost after 31 seconds in Figure 4.5, but after further investigation, the secondary code has switched once more back to the first secondary code. With this code transition in the second data set, further receiver improvements can be implemented once the secondary code pattern is determined so that there is no loss of the data message when the codes change.

4.2.4 Bit Synchronization Approach.

After the initial tracking was completed, the receiver was configured to attempt bit synchronization of the Compass E2 signal. The same approach used in Section 3.6.1 was also used for Compass. The length of which the synchronization was processed was set to 6 s since this was the length of data burst cycle. The same count threshold, 15, and bit ratio, 3, were used to determine if there was successful bit synchronization. Figure 4.6 shows the resulting histogram from the first data set. Compared to the

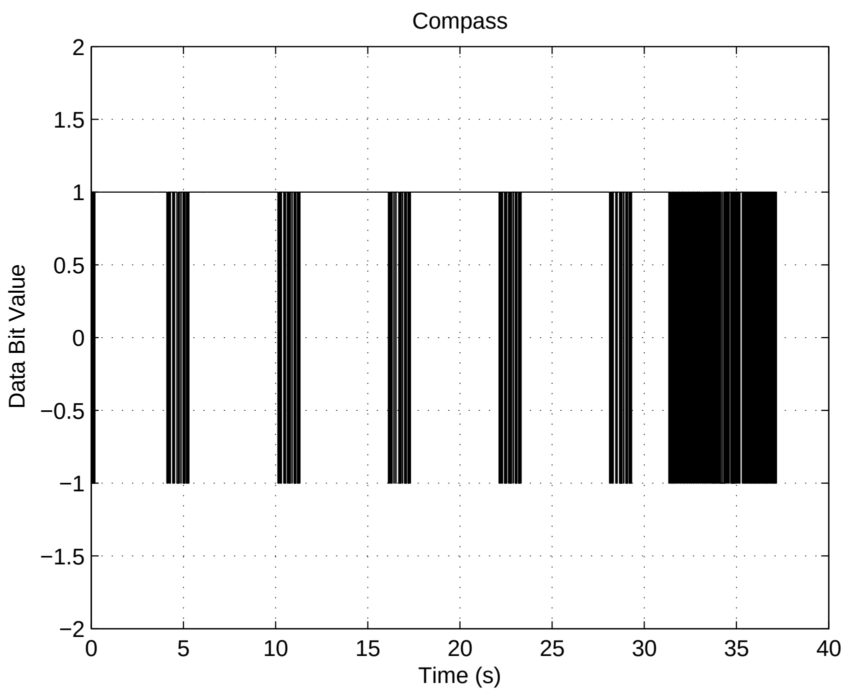


Figure 4.5. Compass' Data Message for Data Set 2. Note the 6 s cycle with approximately a 1.2 s data burst followed by 5 s of no data.

GPS example in Section 3.6.1, there are many more bit flips in other bins. However, this is due to the fact that this data set had a data message burst instead of a large gap at the beginning of the collection.

4.2.5 PIT & B_n Changes.

After proper bit synchronization occurred, the PIT can be increased just as it was with the GPS portion of the receiver. However, the B_{n_D} was held at 5 Hz to allow the PIT to increase to 20 ms. If the B_{n_D} was decreased from 5 Hz, the DLL discriminator values would diverge from 0. This was surprising to find since the DLL is fairly robust to noise when the signal is locked. More research into this phenomenon needs to be studied. Table 4.2 highlights the changes in PIT and B_{n_P} as the data is processed by the receiver.

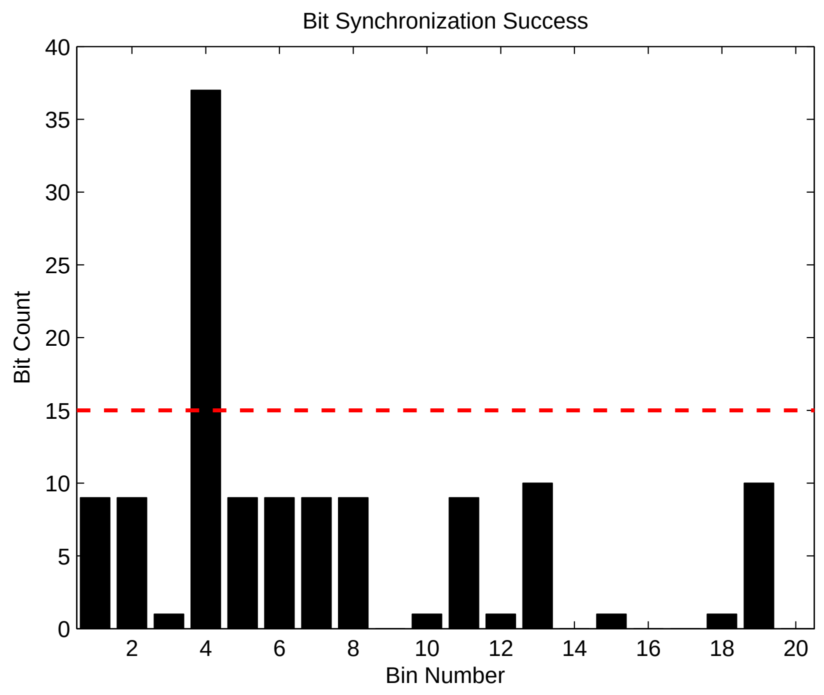


Figure 4.6. Histogram of a Compass successful bit synchronization. As can be clearly seen, the number of data bit transitions exceeds the threshold of 15, and the bit ratio is greater than 3.

After the receiver was modified to correctly change the PIT and B_ns, the receiver was able to properly track the satellite signal. The Doppler shifts for each data set are shown in Figure 4.7. Figure 4.8 shows the output of the PLL discriminator, and Figure 4.9 displays the DLL discriminator output. Figure 4.10 shows the time-domain scatter plot of the Compass E2 signal from Data Set 1. Note the different concentrations of points due to the increasing PIT length.

4.2.6 C/N_0 Calculations.

Compass C/N_0 estimates were calculated the same way as GPS as described in Section 3.6.3. Figure 4.11 shows the calculated C/N_0 values for Compass from both Data Sets. Both plots only start data after the initial 6 s bit synchronization portion has been completed. Notice that the C/N_0 values drop-off in Data Set 2 due to the

Table 4.2. Summary of Compass PIT and B_n changes. Note that B_{n_f} is not included because the FLL is turned off after 120 ms of data is processed.

Time (s)	*PIT (ms)*	B_{n_P} *(Hz)*	B_{n_D} *(Hz)*
0	1	20	5
6	2	15	5
6.2	4	12	5
6.4	20	10	5

unexpected change in the secondary code. The estimates are within a reasonable range for tracking and are comparable to the GPS estimates in Figures 3.16 and 3.17.

4.3 Galileo GNSS SDR Modifications

This section describes the modification made to the current software receiver in order to acquire and track the Galileo L1 signal transmitted from the Galileo In Orbit Validation Element-A (GIOVE-A). The L1 signal is composed of two channels. The first channel, designated L1-B, is the data channel and is a 4 ms long code used for data message transmission. The second channel, L1-C, is a pilot channel and is a longer 8-ms code with a secondary code consisting of 25 bits which lengthens the period to 200 ms [10, 3]. Also, the Galileo L1 signals use Binary Offset Carrier (BOC) modulation which will be further described in Section 4.3.2. For the purpose of tracking and data message decoding, the receiver was only designed for the L1-B signal.

4.3.1 Galileo PRN Generation.

In order to properly acquire and track the L1-B signal, the 4092 chip length PRN code needed to be properly generated. The PRN is a 13-stage Gold code. Previous work from Stanford provided the polynomials and initial states for code generation [10]. These polynomials and states are listed in Table 4.3. A MATLAB

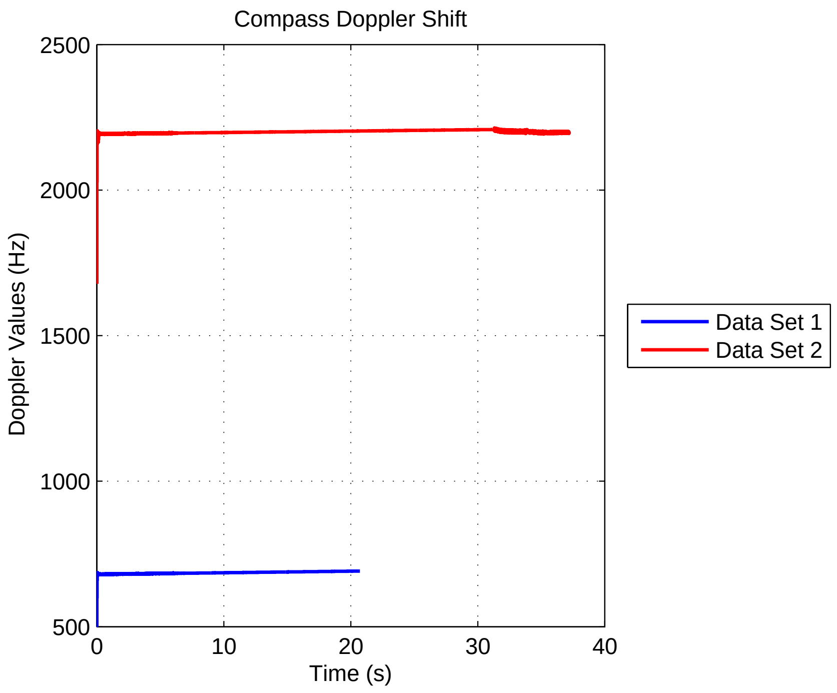


Figure 4.7. Compass Doppler Shifts for Both Data Sets. Note that lock is lost on Data Set 2 after 31 seconds due to the change in the secondary code.

script was written to generate the proper PRN code for the L1-B signal. Once the correct PRN was generated, the BOC modulation needed to be removed from the signal.

Table 4.3. GIOVE-A L1-B 13-stage Gold Code Polynomials and Initial States[10]. These polynomials produce the 4092 chip, 4 ms PRN code.

Polynomial 1	$X^{13}+X^{10}+X^9+X^7+X^5+X^4+1$
Initial State 1	[1 1 1 1 1 1 1 1 1 1 1 1 1]
Polynomial 2	$X^{13}+X^{12}+X^8+X^7+X^6+X^5+1$
Initial State 2	[1 1 0 1 1 1 0 0 0 0 0 1 1]

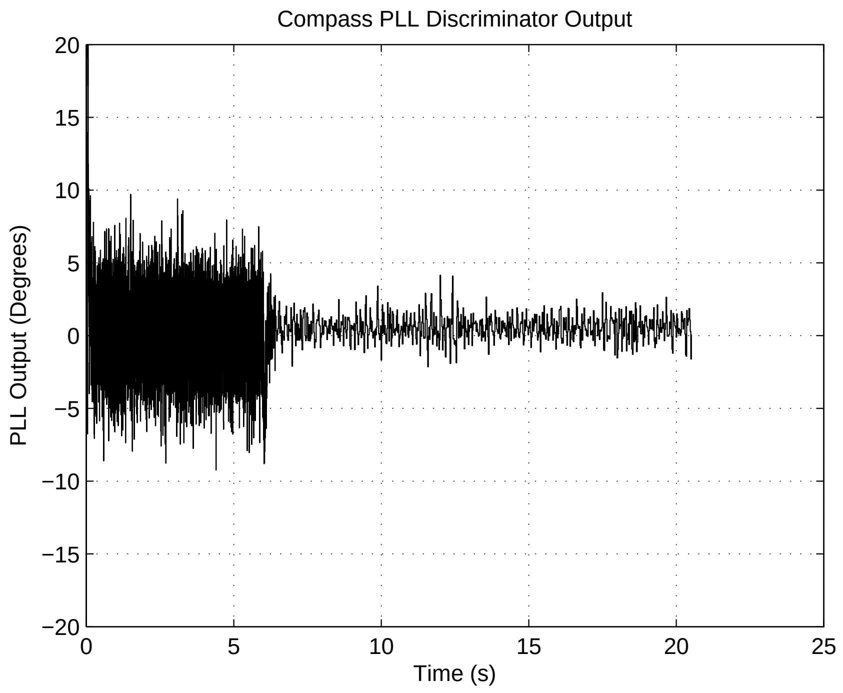


Figure 4.8. Compass PLL Discriminator Output for Data Set 1.

4.3.2 Binary Offset Carrier.

BOC is a type of modulation that spreads the signal into two main lobes around the carrier frequency. This is done by mixing the signal with another square wave. BOC(a, b) is commonly-used BOC notation and indicates the carrier frequency of the BOC is $a \times 1.023$ MHz and the spreading code chipping rate is $b \times 1.023$ MHz [6]. The L1-B channel is a BOC(1,1) modulation [3]. Equation (30) shows the structure of the L1-B signal as transmitted from the satellite,

$$s_{L1-B}(t) = \frac{A}{\sqrt{2}} SC(t) C_{L1-B}(t) D(t) \cos(2\pi f_c t + \phi) \tag{30}$$

where A is the signal amplitude, $SC(t)$ is the BOC(1,1) modulated spreading waveform, $C_{L1-B}(t)$ is the L1-B PRN coded waveform, $D(t)$ is the 250 bit/s data message and $f_c = 1575.42$ MHz is the L1 carrier frequency.

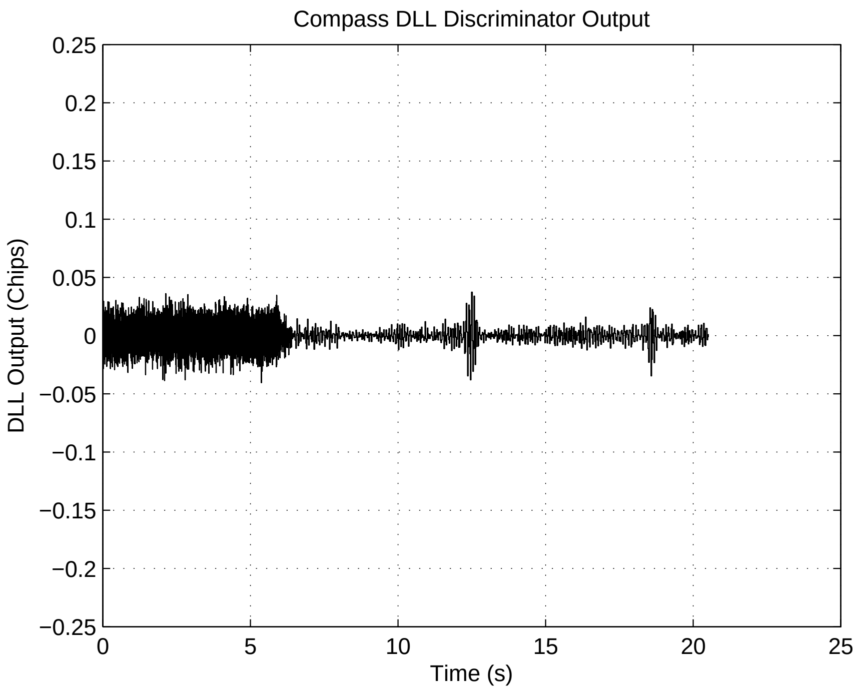


Figure 4.9. Compass DLL Discriminator Output for Data Set 1.

4.3.3 Galileo Signal Acquisition.

The acquisition process for Galileo is similar to that of GPS and Compass. However, since a Galileo data bit is 4 ms in length, the PIT is also 4 ms. Therefore a larger code phase needs to be searched. Since the PIT is 4 ms, this also increases the number of Doppler bins since the size of each Doppler bin decreased to 167 Hz based on Equation (5) from Section 2.5.1.1. This increases the computing time of the algorithm as does the fact that the samples, like Compass' samples, cannot be averaged the same as GPS due to the wider bandwidth of the signal. Figure 4.12 shows the peak of the L1-B signal Data Set 1.

4.3.4 Tracking Initialization.

The tracking of the GIOVE-A L1-B signal was successfully accomplished by making minor changes to the current receiver. The PIT is left at a constant 4 ms because

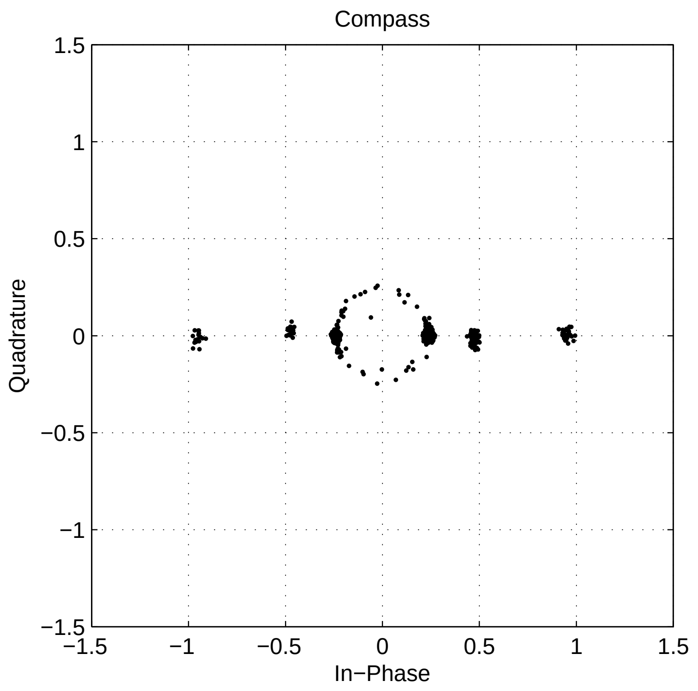


Figure 4.10. Compass Time Domain Scatter Plot. This illustrates that all the data message is in the In-Phase portion of the signal. The tight groupings indicate a strong lock on the signal. Note that the points start near the center and then increase due to the increasing PIT. The points that are off of the I-axis occurred during initial pull-in.

the data message bits are transmitted at 250 bits/s, which equates to 4 ms. Due to this fact, the receiver was coded to start the tracking process at the first full PRN epoch. As done with GPS and Compass, the receiver always processed the data in 1 ms blocks, but then for Galileo, the correlator outputs were then summed over 4 ms intervals corresponding to the full PRN code length epochs.

4.3.5 Correlation Differences.

Next, since the signal has the BOC(1,1) modulation, this is wiped off during the correlation phase of the receiver. The BOC modulation also changes the autocorrelation plot to include two smaller peaks along with the much larger main peak. Figure 4.13 shows an example of this function's output. Therefore, in addition to the

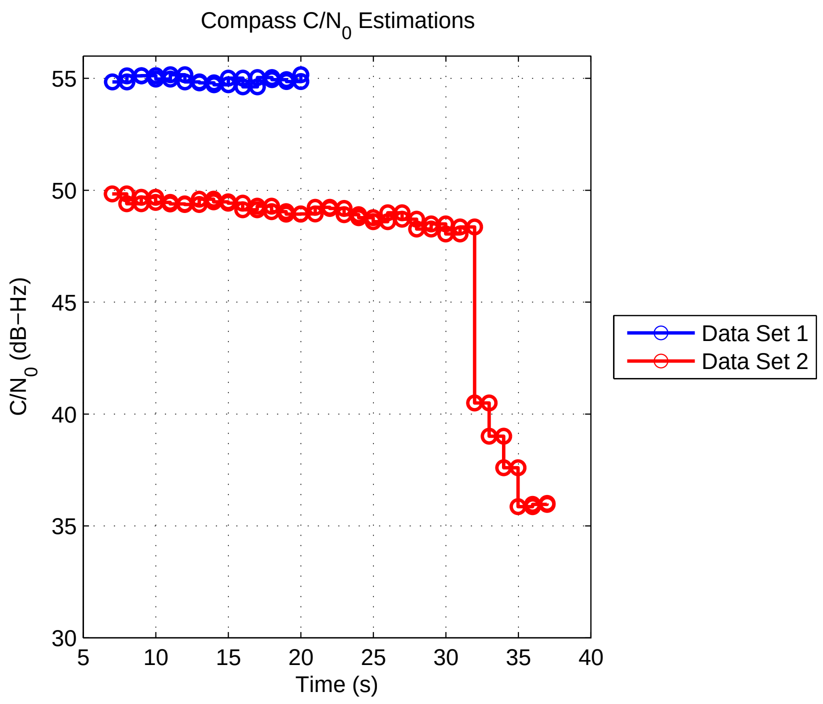


Figure 4.11. Compass C/N_0 Estimates. It must be noted that the drop-off in the C/N_0 values from Data Set 2 occurred due to the fact that the secondary code switched and was not accounted for by the receiver since the code pattern is unknown.

I_E, I_P, I_L, Q_E, Q_P and Q_L correlators, two additional correlators for each channel are added to ensure the tracking of the proper peak. This correlators are referred to as Very Early and Very Late and produce I_{VE}, I_{VL}, Q_{VE}, and Q_{VL} outputs. In order to properly track the BOC signal, the main peak should always be greater than the other two peaks, and the very early and very late outputs should be nearly equal to each other and less than the other three outputs. Figure 4.14 shows a plot of the correlator outputs to illustrate the difference in peak values for these correlators.

4.3.6 4 ms PIT Tracking Considerations.

Since the PRN code and data bit are both 4 ms in length, there are special considerations for tracking the GIOVE-A L1-B signal. As previously mentioned in Section 4.3.4, the initialization of the tracking loop needs to be properly synchronized

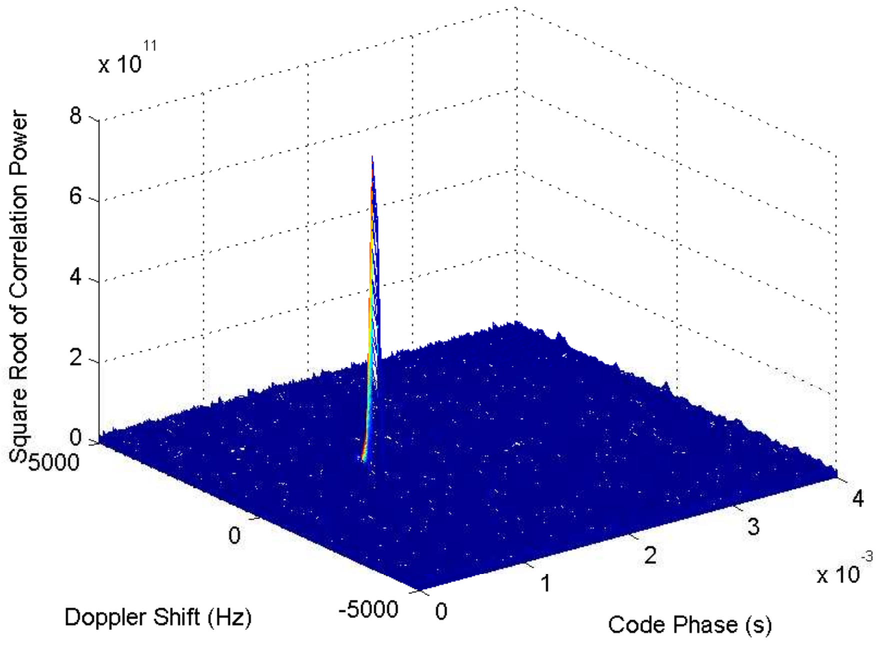


Figure 4.12. Successful Acquisition of GIOVE-A Signal

to the beginning of a data bit. Next, the B_n values are set at $B_{n_F} = 20$ Hz, $B_{n_P} = 12$ Hz and $B_{n_D} = 0.25$ Hz and held constant throughout the signal processing. Finally, due to the PIT length being the same length of a data bit, there is no need for bit synchronization. This is a benefit of the Galileo signal structure. After these modifications were made, the signal was easily tracked. The Doppler Shifts for each run are presented in Figure 4.15. Figures 4.16 and 4.17 show the PLL and DLL discriminator outputs, respectively. A time-domain scatter plot of the In-Phase versus Quadrature arms of the signal is presented in Figure 4.18. Notice that the groupings do not change distances unlike the similar GPS and Compass scatter plots. This is due to the constant PIT of 4 ms. Finally, a portion of the final data bits are highlighted in Figure 4.19.

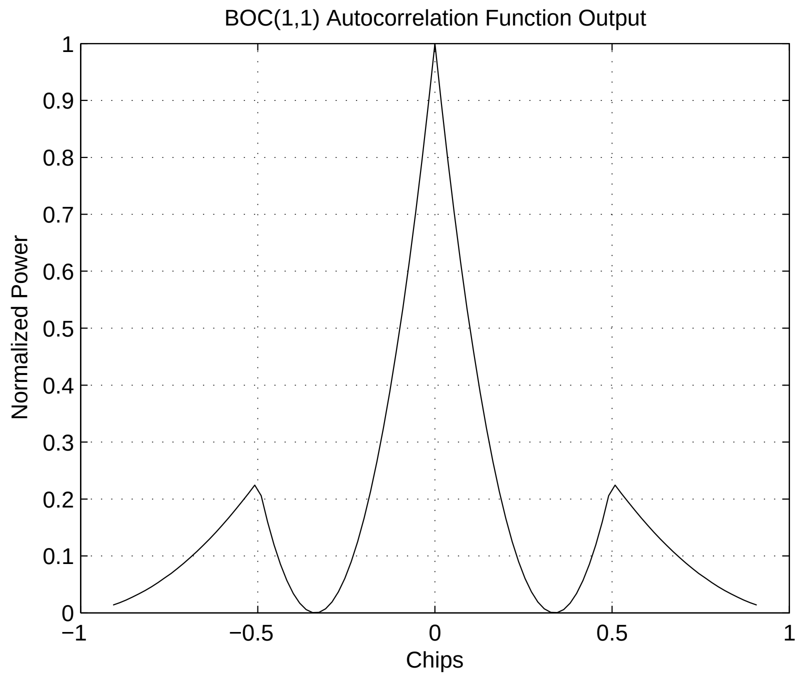


Figure 4.13. BOC(1,1) Autocorrelation Function

4.3.7 C/N_0 Calculations.

The C/N_0 values for the E2 L1-B signal were calculated using the same equations for GPS and Compass. However, since the Galileo data bit length is 4 ms compared to the 20 ms data bit length for GPS and Compass, the M in Equation (28) is changed to 4. Figure 4.20 displays the C/N_0 values for both data sets. The values are approximately 10-15 dB lower compared to the GPS and Compass values. However, the signal strength is still strong enough to enable tracking which has been demonstrated through the previous sections. More research will need to be accomplished to determine why these levels are lower.

4.4 Summary

This chapter described the modifications made to the GPS-only software receiver to allow tracking of Compass and Galileo satellites. For Compass, the main issue that

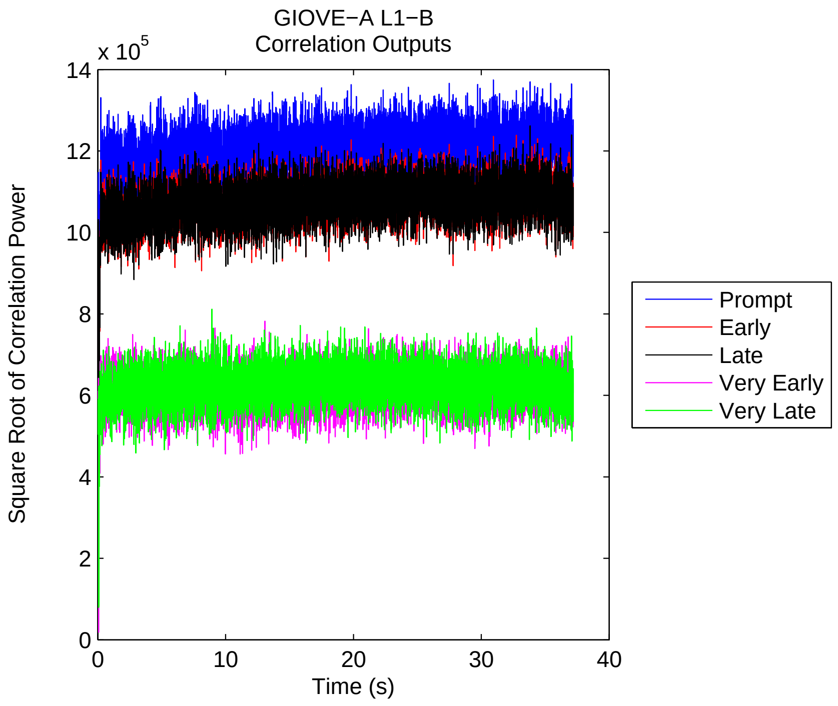


Figure 4.14. GIOVE-A Correlation Outputs. Notice that the Prompt Correlator is the largest value. This indicates that the receiver is properly tracking the GIOVE-A L1-B signal.

was addressed was the addition of a secondary code that neither GPS nor Galileo have. It was also found that there are at least two different secondary codes for Compass. After the secondary codes were determined, the tracking procedure was very similar to the GPS tracking due to the similarities in the primary code structure. The bit synchronization time frame was increased to 6 seconds due to the large gaps that were present in the data output. Finally, the C/N_0 estimations were calculated and were at reasonable levels.

Next, the modifications for the Galileo satellite, GIOVE-A, were presented. The main differences of the longer 4 ms PIT and BOC modulation were discussed. The benefit of the GIOVE-A L1-B signal structure was explained due to the fact that no bit synchronization was needed. Finally, the GIOVE-A L1-B signal's C/N_0 were calculated, but they were 10-15 dB lower than GPS and Compass.

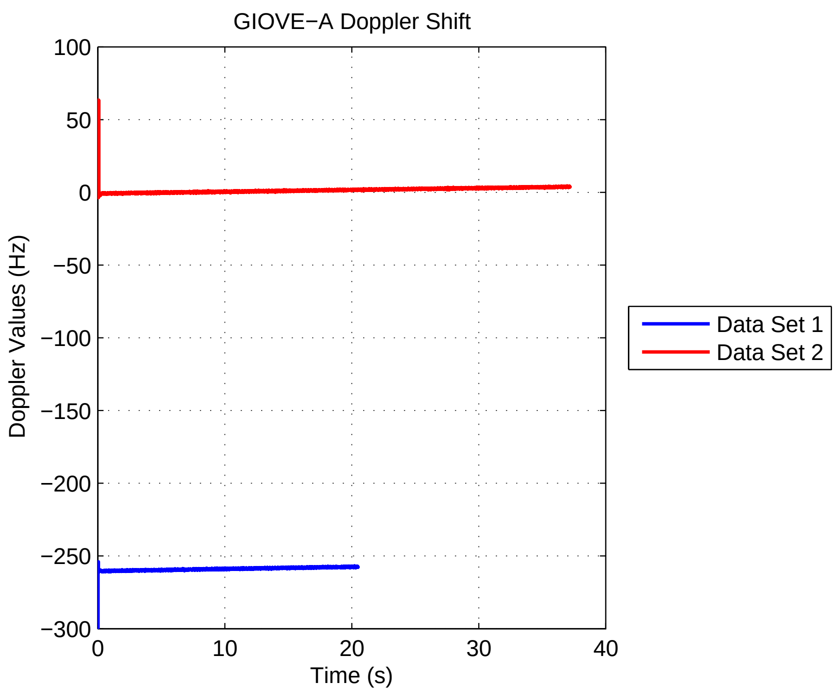


Figure 4.15. GIOVE-A Doppler Shifts for Both Data Sets

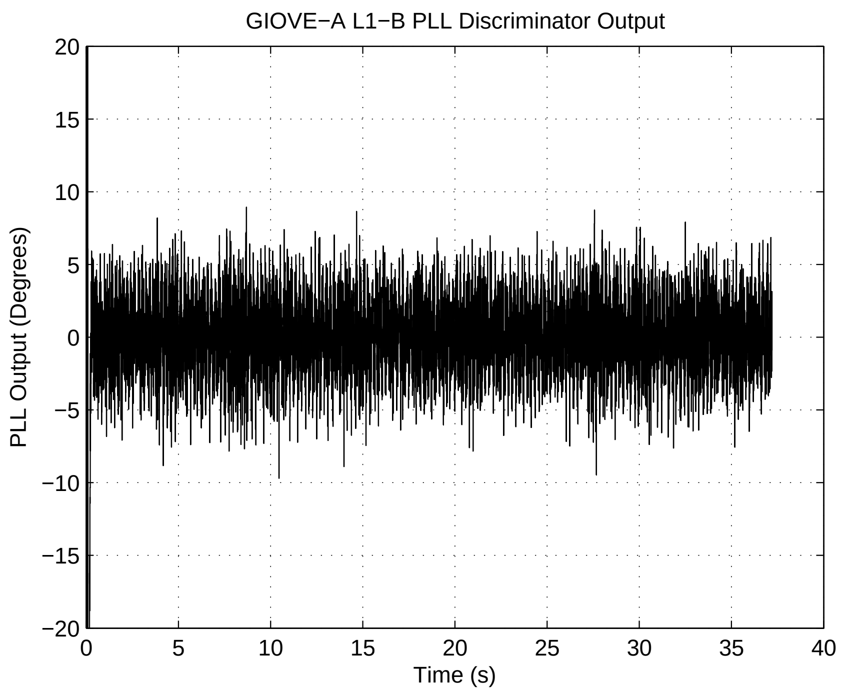


Figure 4.16. GIOVE-A L1-B PLL Discriminator Output from Data Set 2.

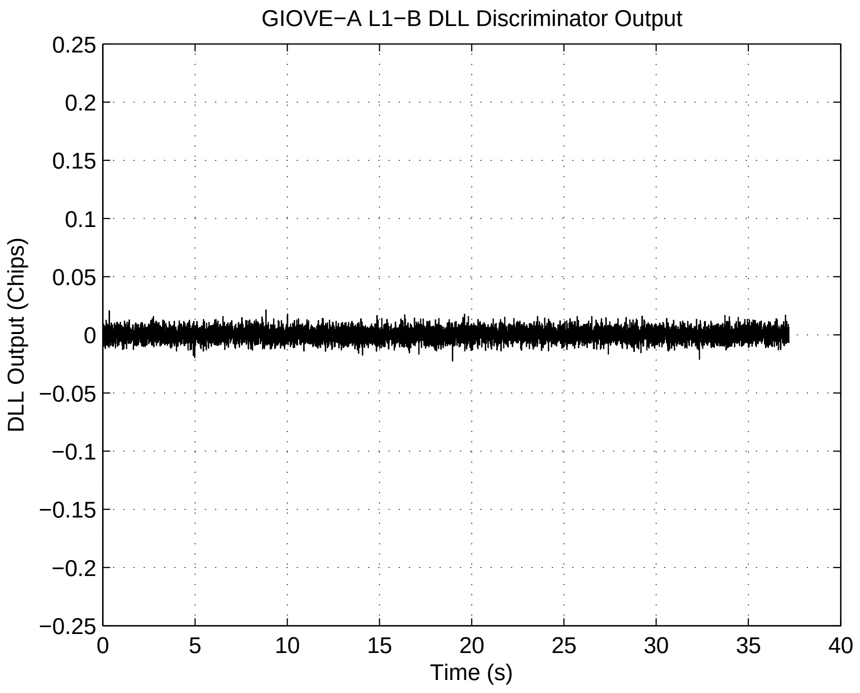


Figure 4.17. GIOVE-A L1-B DLL Discriminator Output from Data Set 2.

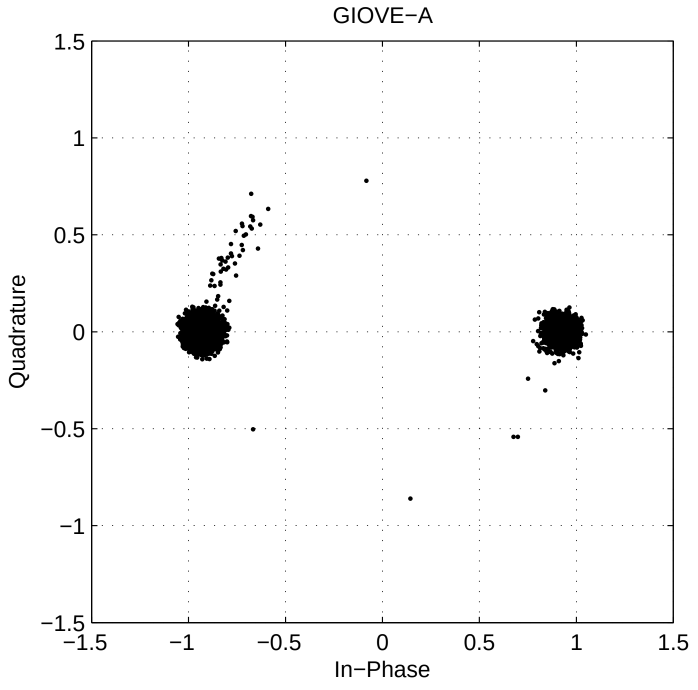


Figure 4.18. GIOVE-A Time Domain Scatter Plot. This illustrates that the data message is in the In-Phase portion of the signal. The points that are off of the I-axis occurred during initial pull-in. The tight groupings indicate a strong lock on the signal. Note that the points stay at the same distance compared to the GPS and Compass scatter plots which have groupings at different distances. This is due to the fact that the GIOVE-A L1-B signal is run with a constant PIT of 4 ms.

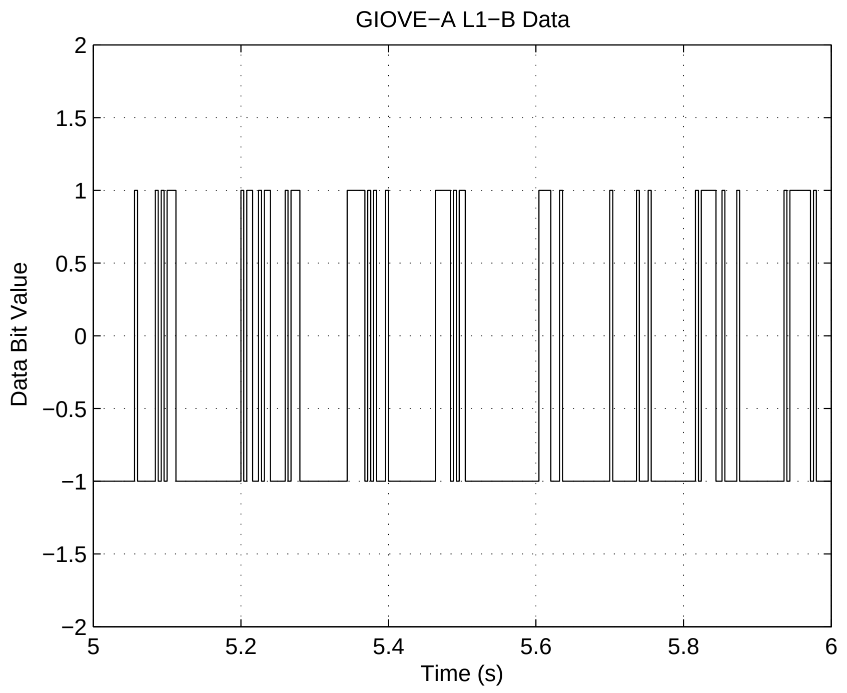


Figure 4.19. One Second Portion of GIOVE-A L1-B Data Message.

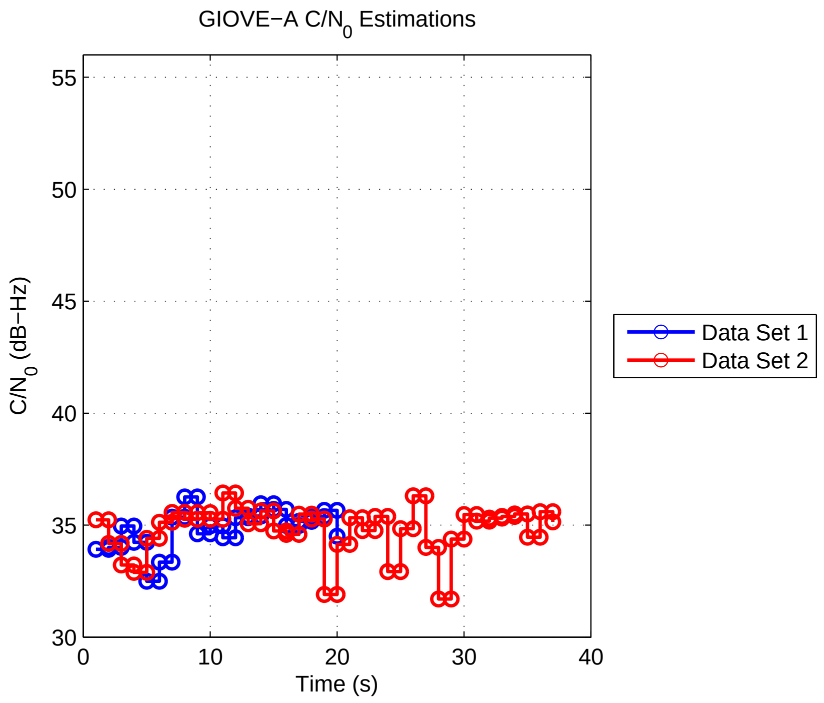


Figure 4.20. GIOVE-A C/N_0 Estimates.

V. Summary and Conclusions

5.1 Summary

This thesis describes an initial GNSS software receiver architecture that AFIT will be using for GNSS receiver research. Due to the growing number of satellites and signals in our skies, this research provides an important first step towards enabling studies of emerging systems and signals. The second chapter provided background information on the major GNSS systems as well as details on how a software receiver generally operates. There are many aspects of the receiver that have to be working properly in order to acquire, track and eventually decode a navigation message. The third chapter explained how the receiver was configured for the United States' GPS system. The satellites were successfully acquired, tracked, decoded and position estimates were calculated. The fourth chapter described the modifications made under this research to the GNSS receiver to allow acquisition and tracking of China's Compass system. Also in this chapter, the modifications for the European Union's Galileo GNSS GIOVE-A satellite were presented to allow proper acquisition and tracking of the L1-B signal.

5.2 Conclusions

Chapter 3 explained how the receiver was set up for the GPS system. The results showed excellent tracking performance since the receiver did not lose lock on the satellites enabling the data message to be output. After the signal was properly tracked, the navigation message was partially decoded in order to calculate pseudorange estimates to allow an estimate of the user's position. The errors in the North and East directions were under 10 m for Data Set 1 and around 30 m for Data Set 2. Since no other corrections were implemented to improve the pseudorange estimates, such as ionospheric or tropospheric errors, these represent reasonable position estimates.

Chapter 4 first described the Compass GNSS modifications. Due to the addition of a secondary code to the primary PRN code, more analysis was performed to discover that there are two secondary codes that are transmitted at different times. Unfortunately with the short duration data sets, the pattern of these codes could not be determined. Until official Compass documentation is published, this pattern may remain unknown. Also, since there is no official documentation the structure of the navigation message is unknown as well. Therefore, the navigation message could not be decoded by this software receiver. After the secondary codes were determined, the tracking process was continued. The Compass discriminator output plots show excellent results for the tracking process.

The latter part of Chapter 4 explained the changes made to the receiver to incorporate the Galileo system's GIOVE-A L1-B signal. The GIOVE-A signal also employs a more modern Binary Offset Carrier (BOC) spreading signal. This modulation spreads the signal into two main peaks to reduce the interference with the GPS L1 C/A code signal at the same carrier frequency. Since the PRN length and navigation bit are both 4 ms in length, this made the PIT a constant 4 ms. This allowed the receiver to start with smaller noise bandwidths. Also, there was no need to perform bit synchronization due to the same PIT and data bit length. Finally, the GIOVE-A results show excellent evidence of successful L1-B signal tracking. Unfortunately, since there are only two Galileo satellites in service, position estimates are unable to be accomplished even though the message can be decoded.

5.3 Further Research & Recommendations

Since the duration of a data collection was limited to the storage size of a standard DVD, longer duration collections will need to be examined once the final hardware is set up at AFIT. This will allow collections at any time and of any size. Once this can be accomplished, full decoding of the GPS and GIOVE-A signal can begin. With the addition of GIOVE-A ephemeris data, the implementation of a GPS/Galileo position estimate may be possible. Another advantage of using longer duration collections is the further studying of the Compass secondary code pattern. If this pattern can be determined, then the receiver can be properly configured to track the signal whenever the satellite is in-view. This will allow uninterrupted research of the navigation message.

In terms of the GNSS software receiver, updates to the code to allow a Graphical User Interface can be easily made. Also, more pseudorange error corrections can be added to the receiver to account for atmospheric propagation or multipath mitigation. Once the receiver is on-site and more frequency bands are able to be studied, the new GPS, GIOVE-B, and future Compass satellites and signals can be incorporated into the receiver. Finally, the biggest improvement that can be made to the receiver is the ability to have it run in real-time. This would be the final goal of the receiver since it would allow real-time studying of signals.

Bibliography

1. *NAVSTAR GPS Space Segment and Navigation User Interfaces.* Fountain Valley, CA, 2000.

2. *GLONASS Interface Control Document, Version 5.0.* Moscow, 2002.

3. *GIOVE-A+B Public SIS ICD*, 2008.

4. "GPS Laboratory - Cornell University", December 2009. `http://gps.ece.cornell.edu/index.html`.

5. Alaqeeli, A., J. Starzyk, and F. van Graas. "Real-time acquisition and tracking for GPS receivers". *Proceedings of the 2003 International Symposium on Circuits and Systems, 2003. ISCAS '03.*, volume 4, IV–500–IV–503 vol.4. May 2003.

6. Borre, Kai, Dennis M. Akos, Nicolaj Bertelsen, Peter Rinder, and Sren Holdt Jensen. *A Sotware-Defind GPS and Galileo Receiver: A Single-Frequency Approach.* Birkhuser Boston, 2007.

7. Charkhandeh, Shahin, M.G. Petovello, R. Watson, and G. Lachapelle. "Implementation and testing of a real-time software-based GPS receiver for x86 processors". *Proceedings of the Institute of Navigation, National Technical Meeting*, volume 2, 927 – 934. Monterey, CA, United States, 2006.

8. De Wilde, Wim, Frank Boon, J-M Sleewaegen, and Frank Wilms. "More Compass Points". 2(5):44–48, July/August 2007.

9. Gao, Grace Xingxin, Alan Chen, Sherman Lo, David De Lorenzo, and Per Enge. "GNSS over China". 2(5):36–43, July/August 2007.

10. Gao, Grace Xingxin, Jim Spilker, Todd Walker, Per Enge, and Anthony R Pratt. "Code Generation Scheme and Property Analysis of Broadcast Galileo L1 and E6 Signals". *ION GNSS-2006.* Fort Worth, TX, United States, 2006.

11. Gibbons, Glen. "GLONASS-A New Look for the 21st Century". 3(4), May/June 2008.

12. Girau, G., A. Tomatis, F. Dovis, and P. Mulassano. "Efficient Software Defined Radio Implementations of GNSS Receivers". *IEEE International Symposium on Circuits and Systems, 2007. ISCAS 2007.*, 1733–1736. May 2007.

13. Gunawardena, Sanjeev. "Personal Correspondence", September 2009.

14. Gunawardena, Sanjeev. "Personal Correspondence", February 2010.

15. Gunawardena, Sanjeev, Frank Van Graas, and Andrey Soloviev. "Real time block processing engine for software GNSS receivers". *Proceedings of the National Technical Meeting, Institute of Navigation*, volume 2004, 371 – 377. San Diego, CA, United States, 2004.

16. Gunawardena, Sanjeev, Zhen Zhu, and Frank Van Graas. "Triple Frequency RF Front-End for GNSS Instrumentation Receiver Applications". *ION GNSS-2008*. Savannah, GA, United States, 2008.

17. Hein, Guenter W., Thomas Pany, Stefan Wallner, and Jong-Hoon Won. "Platforms for a Future GNSS Receiver". 1(2):56–62, March 2006.

18. Hong, Jin Seok, Jung Won Lee, Gyu-In Jee, Young Jae Lee, Jin Won Kim, and Chan Gook Park. "GPS Signal Processing Algorithm for Software GPS Receiver". *ION GPS-2000*. Salt Lake City, UT, United States, 2000.

19. Jovancevic, Aleksandar, Andrew Brown, Suman Ganguly, Jignesh Goda, Michael Kirchner, and Salvisa Zigic. "Real-Time Dual Frequency Software Receiver". *Proceedings of the 16th International Technical Meeting of the Satellite Division of the Institute of Navigation*, volume 2003. Portland, OR, United States, 2003.

20. Li, Zuohu, Jinming Hao, Jianwen Li, and Chengjun Zhang. "IF Signal Processing Algorithm for GPS Software Receiver". *Congress on Image and Signal Processing, 2008. CISP '08.*, volume 1, 160–164. May 2008.

21. Misra, Pratap and Per Enge. *Global Positioning System: Signals, Measurements, and Performance*. Ganga-Jamuna Press, second edition, 2006.

22. Mitola, J. "The Software Radio Architecture". 33(5):26–38, May 1995. ISSN 0163-6804.

23. Molino, Andrea, Gianmarco Girau, Mario Nicola, Maurizio Fantino, and Marco Pini. "Evaluation of a FFT-Based Acquisition in Real Time Hardware and Software GNSS Receivers". *IEEE 10th International Symposium on Spread Spectrum Techniques and Applications, 2008. ISSSTA '08.*, 32–36. Aug. 2008.

24. Qingxi, Zeng, Wang Qing, Pan Shuguo, and Li Chuanjun. "A GPS L1 Software Receiver Implementation on a DSP Platform". *First International Conference on Intelligent Networks and Intelligent Systems, 2008. ICINIS '08.*, 612–615. Nov. 2008.

25. Raquet, John F. "GPS Receiver Design Fundamentals", June 2009. Presented at 2009 Workshop of the Consortium of Ohio Universities for Navigation and Timekeeping (COUNT).

26. Sun, Chih-Cheng and Shau-Shiun Jan. "GNSS signal acquisition and tracking using a parallel approach". *Position Location and Navigation Symposium, 2008 IEEE/ION*, 1332–1340. May 2008.

27. Van Dierendonck, A. J. "GPS Receivers". Bradford W. Parkinson and James J. Spilker Jr. (editors), *Global Positioning System: Theory and Application*, volume 1. American Institue of Aeronautics and Astronautics, 1996.

28. Van Nee, D.J.R. and A.J.R.M. Coenen. "New Fast GPS code-acquisition technique using FFT". 27(2):158–160, Jan. 1991. ISSN 0013-5194.

29. Ward, Phillip. "Satellite Signal Acquisition and Tracking". Elliot D. Kaplan (editor), *Understanding GPS: Principles and Applications*. Artech House Boston, 1996.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704–0188*

1. REPORT DATE *(DD–MM–YYYY)* 25–03–2010	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From — To)* Aug 2008 — Mar 2010

4. TITLE AND SUBTITLE Global Navigation Satellite System Software Defined Radio	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Jason M. McGinthy, Capt, USAF	**5d. PROJECT NUMBER** 10-328
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT/GE/ENG/10-17
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFSPC GPSW/EN (Col David Goldstein) 483 N. Aviation Blvd. Los Angeles AFB El Segundo, CA 90245 633-3226, David.Goldstein@losangeles.af.mil	**10. SPONSOR/MONITOR'S ACRONYM(S)** GPSW/EN **11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

12. DISTRIBUTION / AVAILABILITY STATEMENT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

There are two major trends that are driving the nature of GNSS receiver development today. First, there is a growing number of Global Navigation Satellite Systems (GNSS) that exist or are under development, so that in the near future, users will potentially have access to many more satellites and systems than they do today. The second major trend is the growth in computing power and flexible signal processing capability. These two factors, combined together, point to the value of having a software-defined radio that is able to acquire and track as many GNSS signals, from various systems, as possible. This research describes the development of a GNSS software receiver at the Advanced Navigation Technology (ANT) Center at AFIT. This software receiver uses data collected by a high-quality receiver front end developed by Ohio University, which has a 24 MHz bandwidth and a 12-bit sampler operating at 57M samples per second. The ANT Center Software Receiver currently runs in a post-processing mode, and is able to simultaneously acquire and track signals from multiple GNSS systems.

15. SUBJECT TERMS

Global Navigation Satellite System, GPS, Software Defined Radio, Software Receiver

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dr. John F. Raquet
U	U	U	U	99	**19b. TELEPHONE NUMBER** *(include area code)* (937) 255-3636, x4580; john.raquet@afit.edu